TUGAS AKHIR – KI091391

DETEKSI CITRA RETINA MATA UNTUK PROLIFERATIVE DIABETIC RETINOPATHY MENGGUNAKAN SELEKSI FITUR GENETIC ALGORITHM DAN SUPPORT VECTOR MACHINE

RIZKIFIKA ASANUL IN'AM
5112100121

Dosen Pembimbing I
Dr. Eng. Chastine Fatichah, S.Kom., M.Kom.

Dosen Pembimbing II
Rully Soelaiman, S.Kom., M.Kom.

JURUSAN TEKNIK INFORMATIKA
Fakultas Teknologi Informasi
Institut Teknologi Sepuluh Nopember
Surabaya, 2016

TUGAS AKHIR – KI091391

DETEKSI CITRA RETINA MATA UNTUK PROLIFERATIVE DIABETIC RETINOPATHY MENGGUNAKAN SELEKSI FITUR GENETIC ALGORITHM DAN SUPPORT VECTOR MACHINE

RIZKIFIKA ASANUL IN'AM
5112100121

Dosen Pembimbing I
Dr. Eng. Chastine Fatichah, S.Kom., M.Kom.

Dosen Pembimbing II
Rully Soelaiman, S.Kom, M.Kom.

JURUSAN TEKNIK INFORMATIKA
Fakultas Teknologi Informasi
Institut Teknologi Sepuluh Nopember
Surabaya, 2016

[Halaman ini sengaja dikosongkan]

FINAL PROJECT - KI091391

RETINA IMAGE DETECTION FOR PROLIVERATIVE DIABETIC RETINOPATHY USING GENETIC ALGORITHM AS FEATURE SELECTION AND SUPPORT VECTOR MACHINE

RIZKIFIKA ASANUL IN'AM
5112100121

Supervisor
Dr. Eng. Chastine Fatichah, S.Kom., M.Kom.
Rully Soelaiman, S.Kom, M.Kom.

DEPARTMENT OF INFORMATICS
FACULTY OF INFORMATION TECHNOLOGY
Sepuluh Nopember Institute of Technology
Surabaya, 2016

[Halaman ini sengaja dikosongkan]

LEMBAR PENGESAHAN

DETEKSI CITRA RETINA MATA UNTUK PROLIFERATIVE DIABETIC RETINOPATHY MENGGUNAKAN SELEKSI FITUR GENETIC ALGORITHM DAN SUPPORT VECTOR MACHINE TUGAS AKHIR

Diajukan Untuk Memenuhi Salah Satu Syarat
Memperoleh Gelar Sarjana Komputer
Pada
Bidang Studi Komputasi Cerdas Visual
Program Studi S-1 Jurusan Teknik Informatika
Fakulatas Teknologi Informasi
Institut Teknologi Sepuluh Nopember

Oleh
RIZKIFIKA ASANUL IN'AM
NRP: 5112 100 121

Disetujui oleh Dosen Pembimbing Tugas Akhir:

1. Dr. Eng. Chastine Fatichah, S.Kom., M.Kom. (Pembimbing 1)
   NIP: 197512202001122002

2. Rully Soelaiman, S.Kom., M.Kom. (Pembimbing 2)
   NIP: 197002131994021001

SURABAYA
JUNI 2016

[Halaman ini sengaja dikosongkan]

DETEKSI CITRA RETINA MATA UNTUK *PROLIFERATIVE DIABETIC RETINOPATHY* MENGGUNAKAN SELEKSI FITUR *GENETIC ALGORITHM* DAN *SUPPORT VECTOR MACHINE*

Nama Mahasiswa	**: Rizkifika Asanul In'am**
NRP	**: 5112 100 121**
Jurusan	**: Teknik Informatika FTIF ITS**
Dosen Pembimbing 1	**: Dr. Eng. Chastine Fatichah, S.Kom., M.Kom.**
Dosen Pembimbing 2	**: Rully Soelaiman, S.Kom., M.Kom.**

Abstrak

Diabetes militus adalah gangguan dari peredaran darah manusia. Diabetes ini disebabkan akan kurangnya insulin di dalam darah. Hal ini dapat menyebabkan meningkatnya tekanan darah yang dapat menyebabkan kerusakan sel pada jaringan tubuh dikarenakan tingginya tekanan darah. Salah satu akibat dari diabetes ini dapat dilihat melalui pembengkakan pembuluh pada jaringan organ. Salah satu jaringan yang dapat terganggu dari pembengkakan pembuluh darah adalah mata. Pada retina mata yang mengalami pembengkakan dapat disebut proliferative diabetic retinopathy. Untuk mempermudah pendeteksian dilakukan proses segmentasi vaskular, ekstraksi fitur segmen, serta klasifikasi.

Tugas akhir ini mengimplementasikan salah satu metode pengklasifikasi yang dapat digunakan untuk klasifikasi segmen vaskular retina yaitu Support Vector Machine dan menggunakan metode seleksi fitur Genetic Algorithm. Tahap pertama yakni melakukan perbaikan terhadap citra menggunakan Frangi filter, metode praproses menggunakan mean filter dan masking, serta segmentasi citra menggunakan Bradley thresholding. Setelah itu dilakukan ekstraksi fitur untuk menjadi bahan inputan Genetic Algorithm yang dipadukan dengan Support Vector Machine.

Hasil klasifikasi segmen vaskular normal dan abnormal dengan citra retina dari basis data STARE menggunakan metode Support Vector Machine yang dipadukan dengan seleksi fitur Genetic Algorithm menunjukkan akurasi terbaik sebesar 100% dengan menggunakan konfigurasi kernel polynomial dan method quadratic programming. Sehingga dapat disimpulkan bahwa kalsifikasi segmentasi vaskular retina yang digunakan pada Tugas akhir ini mampu melakukan segmentasi dan klasifikasi dengan baik.

Kata kunci: Klasifikasi, Genentic Algorithm, Support Vector Machine, Vaskular Retina.

RETINA IMAGE DETECTION FOR PROLIVERATIVE DIABETIC RETINOPATHY USING GENETIC ALGORITHM AS FEATURE SELECTION AND SUPPORT VECTOR MACHINE

Student's Name : Rizkifika Asanul In'am
Student's ID : 5112 100 121
Department : Informatics Engineering, FTIF-ITS
First Advisor : Dr. Eng. Chastine Fatichah, S.Kom., M.Kom.
Second Advisor : Rully Soelaiman, S.Kom., M.Kom.

Abstract

Diabetes militus is a diasese of human blood circulation. Diabetes is caused by lack amount of insulin in blood. It cause the higher blood pressure that can lead to vessel blood damage. One of the symtomps that can be seen is in retinal blood vessel. This kind of damage can be called proliferative diabetic retinopathy (PDR). To make it easier to detect PDR in retinal vessel we can detect it by doing segmentation of retinal vessel, feature exstraction of segmented vessel, and classification.

The final project is to implement one of the methods classifiers that can be used for classification of retinal vascular segments namely Support Vector Machine and Genetic Algorithm as feature selection to optimize the classifier. The first step is to do image correction by using Frangi filter, Mean filter, and masking. The result is then segmented by using Bradley thresholding. The image is extracted to get some information related called feature. The feature is optimized by Genetic Algorithm collaborated with Support Vector Machine.

The results of the classification of normal and abnormal vascular segments by retinal image from STARE database using Support Vector Machine methods shows the best accuracy of 100%

using kernel type polynomial and mthod quadratic programming. Therefor it can be concluded that the method used for retinal vascular segmentation and retinal vascular segments classification in this final project is reliable for segmentation and classification.

Keywords: Classification, Genetic Algorithm, Support Vector Machine, Retinal Vascular.

KATA PENGANTAR

بِسْمِ ٱللَّهِ ٱلرَّحْمَٰنِ ٱلرَّحِيمِ

Alhamdulillahirabbil'alamin, segala puji bagi Allah SWT, yang telah melimpahkan rahmat dan hidayah-Nya sehingga penulis dapat menyelesaikan Tugas Akhir yang berjudul ***"DETEKSI CITRA RETINA MATA UNTUK PROLIFERATIVE DIABETIC RETINOPATHY MENGGUNAKAN SELEKSI FITUR GENETIC ALGORITHM DAN BERBASIS DUAL CLASSIFICATION SUPPORT VECTOR MACHINE".***

Pengerjaan Tugas Akhir ini merupakan suatu kesempatan yang sangat baik bagi penulis. Dengan pengerjaan Tugas Akhir ini, penulis bisa belajar lebih banyak untuk memperdalam dan meningkatkan apa yang telah didapatkan penulis selama menempuh perkuliahan di Teknik Informatika ITS. Dengan Tugas Akhir ini penulis juga dapat menghasilkan suatu implementasi dari apa yang telah penulis pelajari.

Selesainya Tugas Akhir ini tidak lepas dari bantuan dan dukungan beberapa pihak. Sehingga pada kesempatan ini penulis mengucapkan terima kasih kepada:

1. Bapak Rully Soelaiman, S.Kom., M.Kom. dan Ibu Dr. Eng. Chastine Fatichah, S.Kom., M.Kom. selaku pembimbing yang telah memberikan motivasi, nasehat, bimbingan dan bantuan kepada penulis sehingga penulis dapat menyelesaikan Tugas Akhir ini.
2. Ayah dan Ibu dari penulis, Bapak Purwo Setyo Sugondo, Ibu Endang Kurniati, dan Ibu Siti Nur Hidayati, serta saudara penulis yang senantiasa memberikan dukungan moral, material, serta doa. Serta menjadi motivasi penulis untuk selalu menjalankan kuliah dengan baik dan menyelesaikan Tugas Akhir ini.
3. Segenap dosen Teknik Informatika yang telah memberikan ilmunya selama penulis berkuliah di Teknik Informatika ITS.

4. Para penulis artikel ilmiah yang digunakan pada Tugas akhir ini, pendiri Google, pendiri Mathworks, serta berbagai referensi lain, yang telah membantu mempermudah penyelesaian Tugas Akhir ini.
5. Kawan-kawan angkatan 2012 yang menemani, membantu, dan memotivasi selama penulis menjalankan seluruh tugas perkuliahan hingga penulis sampai pada tahap pengerjaan Tugas Akhir ini.
6. Serta semua pihak yang yang telah turut membantu penulis dalam menyelesaikan Tugas Akhir ini.

Penulis menyadari bahwa Tugas Akhir ini masih memiliki banyak kekurangan. Sehingga dengan kerendahan hati, penulis mengharapkan kritik dan saran dari pembaca untuk perbaikan ke depan.

Surabaya, Juni 2016

Rizkifika Asanul In'am

DAFTAR ISI

LEMBAR PENGESAHAN v
Abstrak vii
Abstract ix
KATA PENGANTAR xi
DAFTAR ISI xiii
DAFTAR GAMBAR xvii
DAFTAR KODE SUMBER xxvii
1 BAB I PENDAHULUAN 1
1.1. Latar Belakang 1
1.2. Rumusan Permasalahan 2
1.3. Batasan Masalah 2
1.4. Tujuan dan Manfaat 3
1.5. Metodologi 3
1.6. Sistematika Laporan 5
2 BAB II TINJAUAN PUSTAKA 7
2.1. Praproses Citra Retina 7
2.1.1 *Masking* 8
2.1.2 *Green Channel* 8
2.2. Segmentasi Vaskular Retina 9
2.2.1 *Hessian Matrix* 9
2.2.2 *Eigenvalue* dan Vektor 10
2.2.3 *Frangi Filter* 10
2.2.4 *Mean Filter* 11
2.2.5 *Bradley Local Image Thresholding* 12
2.2.6 *Morphological Cleaning* dan *Image Enhancement* 13
2.3. Ekstraksi Fitur Vaskular Retina 13
2.3.1 *Pixel Area* 14
2.3.2 *Energy* 14
2.3.3 *Mean Gradient* 14
2.3.4 *Standard Deviation Gradient* 15
2.3.5 *Mean Grayscale* 15

2.3.6 *Vessel Segment* 15
2.3.7 *Vessel Width Mean* 15
2.3.8 *Vessel Length Mean* 16
2.4. Seleksi Fitur *Genetic Algorithm* 16
2.4.1 Proses Utama *Genetic Algorithm* 16
2.4.2 *Fitness Function* 18
2.5. *Support Vector Machine* 19
2.6. *Kernel Trick* 20
3 BAB III PERANCANGAN 21
3.1. Perancangan Praproses dan Segmentasi Vaskular 21
3.1.1 Program Utama 31
3.1.2 *Hessian Matrix* dan *Eigenvalue* 32
3.1.3 *Frangi Filter* 33
3.1.4 *Mean Filter* 36
3.1.5 *Masking* 36
3.1.6 *Bradley Thresholding* 37
3.1.7 Morfologi *Open* 38
3.1.8 Morfologi *Close* 38
3.2. Perancangan Ekstraksi Fitur Citra Segmentasi Retina 38
3.2.1 Program Utama 43
3.2.2 Normalisasi Fitur 44
3.2.3 Fitur *Area* 45
3.2.4 Fitur *Energy* 45
3.2.5 Fitur *Mean Gradient* 46
3.2.6 Fitur *Standard Deviation Gradient* 47
3.2.7 Fitur *Mean Gray Level* 47
3.2.8 Fitur *Vessel Segment* 48
3.2.9 Fitur *Vessel Length Mean* 48
3.2.10 Fitur *Vessel Width Mean* 49
3.3. Perancangan Seleksi Fitur *Genetic Algorithm* dan Klasifikasi *Support Vector Machine* 50

3.3.1 Program *Crossover* 53
3.3.2 Program *Mutation* 54
3.3.3 Program *CopyChromosome* 54
3.3.4 Program toStr 55
3.3.5 Program *Generate Initial Generation* 56
3.3.6 Program Utama *Genetic Algorithm* dan SVM 56
4 BAB IV IMPLEMENTASI 59
4.1. Lingkungan Implementasi 59
4.2. Implementasi 59
4.2.1. Implementasi Pemindaian Retina Mata 59
4.2.2. Implementasi Praproses dan Segmentasi Citra Retina 60
4.2.3. Implementasi Ekstraksi Fitur Segmen Vaskular Retina 65
4.2.4. Seleksi Fitur *Genetic Algorithm* dan Klasifikasi *Support Vector Machine* 72
5 BAB V UJI COBA DAN EVALUASI 81
5.1. Lingkungan Uji Coba 81
5.2. Uji Coba Kasifikasi Citra 81
5.2.1 Data Uji Coba Klasifikasi Citra 82
5.2.2 Uji Coba dan Evalusi Klasifikasi Berdasarkan Iterasi *Genetic Algorithm* 82
5.2.3 Uji Coba dan Evalusi Klasifikasi Berdasarkan Kernel SVM 83
5.2.4 Uji Coba dan Evalusi Klasifikasi Berdasarkan Method SVM 91
5.2.5 Uji Coba dan Evalusi Klasifikasi Tanpa Genetic Algorithm 95
6 BAB VI KESIMPULAN DAN SARAN 99
6.1. Kesimpulan 99
6.2. Saran 99
7 LAMPIRAN A 103

[Halaman ini sengaja dikosongkan]

DAFTAR GAMBAR

Gambar 2. 1 Citra retina dari basis data STARE [2].....7
Gambar 2. 2 Citra retina dalam *inverse green channel* dan citra hasil praproses 8
Gambar 2. 3 Citra *masking* dan hasil *shrinking* 9
Gambar 2. 4 Citra hasil dari *Frangi filter* 11
Gambar 2. 5 hasil *mean filter* 12
Gambar 2. 6 hasil *Bradley Threshold*.....12
Gambar 2. 7 Citra hasil akhir segmentasi.....13
Gambar 2. 8 Alur GA dan SVM 18
Gambar 2. 9 Ilustrasi *hyperplane* SVM 19

Gambar 3. 1 Gambaran umum proses keseluruhan.....21
Gambar 3. 2 Proses utama praproses dan segmentasi 22
Gambar 3. 3 *Pseudocode* program utama 31
Gambar 3. 4 *Pseudocode* program *Hessian Matrix* 32
Gambar 3. 5 *Psuedocode Eigenvalue*.....33
Gambar 3. 6 *Pseudocode Frangi Filter* (Bagian Pertama) 34
Gambar 3. 7 *Pseudocode Frangi Filter* (Bagian Kedua) 35
Gambar 3. 8 *Pseudocode Mean Filter* 36
Gambar 3. 9 *Pseudocode Masking* (Bagian Pertama).....36
Gambar 3. 10 *Pseudocode Masking* (Bagian Kedua).....37
Gambar 3. 11 *Pseudocode Bradlay Thresholding*.....37
Gambar 3. 12 *Pseudocode Open* (Bagian Pertama).....38
Gambar 3. 13 *Pseudocode close* 38
Gambar 3. 14 Gambaran umum proses ekstraksi fitur segmen vaskular retina 39
Gambar 3. 15 *Pseudocode* program utama ekstraksi fitur (Bagian Satu).....43
Gambar 3. 16 *Pseudocode* program utama ekstraksi fitur (Bagian Kedua) 44
Gambar 3. 17 *Pseudocode* program normalisasi fitur (Bagian Satu) 44

Gambar 3. 18 *Pseudocode* program normalisasi fitur (Bagian Kedua) 45
Gambar 3. 19 *Pseudocode* program menghitung *area* 45
Gambar 3. 20 *Pseudocode* program menghitung *energy* 46
Gambar 3. 21 *Pseudocode* program menghitung *mean gradient* ..46
Gambar 3. 22 *Pseudocode* program menghitung *standard deviation gradient* (bagian pertama) 47
Gambar 3. 23 *Pseudocode Mean Gray Level.* 47
Gambar 3. 24 *Pseudocode Vessel Segment* 48
Gambar 3. 25 . *Pseudocode Vessel Length Mean* (Bagian Pertama) 48
Gambar 3. 26 *Pseudocode Vessel Length Mean* (Bagian Kedua) .49
Gambar 3. 27 *Pseudocode* program menghitung *mean vessel width* (Bagian Pertama) 49
Gambar 3. 28 *Pseudocode* program menghitung mean *vessel width* (Bagian Kedua).....50
Gambar 3. 29 diagram alur GA dan SVM 51
Gambar 3. 30 Desain Kromosom.....51
Gambar 3. 31 *Psuedocode Crossover* 53
Gambar 3. 32 *Psuedocode Mutation* 54
Gambar 3. 33 *Psuedocode Copychromosome* (Bagian Pertama) ..54
Gambar 3. 34 *Psuedocode Copychromosome* (Bagian kedua) 55
Gambar 3. 35 *Psuedocode toStr* 55
Gambar 3. 36 *Pseudocode* program *Initial generation* 56
Gambar 3. 37 *Pseudocode* program utama GA.....57

Gambar 5. 1. Contoh citra FP.....85
Gambar 5. 2 Contoh Citra FN 85
Gambar 5. 3 Contoh citra TN.....86
Gambar 5. 4. Contoh citra FP.....87
Gambar 5. 5 Contoh citra FP.....88
Gambar 5. 6 Contoh Citra FN 88
Gambar 5. 7 Contoh Citra FN 89
Gambar 5. 8 Contoh citra FN 90
Gambar 5. 9 Contoh citra FP.....90

Gambar 5. 10. Contoh citra FN 91
Gambar 5. 11 Contoh citra FN 93
Gambar 5. 12 Contoh citra FP 93
Gambar 5. 13. Contoh citra FP 94
Gambar 5. 14 Contoh citra TN 96
Gambar 5. 15 Contoh citra FN 96
Gambar 5. 16 Contoh citra TN 97
Gambar 5. 17 Contoh citra FN 97
Gambar 5. 18 Contoh citra TN 98

[Halaman ini sengaja dikosongkan]

DAFTAR TABEL

Tabel 2. 1 Trik Kernel SVM yang umum.....20

Tabel 3. 1 Daftar Variabel yang Digunakan Pada *Pseudocode* Praproses dan Segmentasi Pembuluh Darah Retina (Bagian Pertama) 23
Tabel 3. 2 Daftar Variabel yang Digunakan Pada *Pseudocode* Praproses dan Segmentasi Pembuluh Darah Retina (Bagian Kedua) 24
Tabel 3. 3 Daftar Variabel yang Digunakan Pada *Pseudocode* Praproses dan Segmentasi Pembuluh Darah Retina (Bagian Ketiga) 25
Tabel 3. 4 Daftar Variabel yang Digunakan Pada *Pseudocode* Praproses dan Segmentasi Pembuluh Darah Retina (Bagian Pertama) 26
Tabel 3. 5 Daftar Variabel yang Digunakan Pada *Pseudocode* Praproses dan Segmentasi Pembuluh Darah Retina (Bagian Kelima).....27
Tabel 3. 6 Daftar Variabel yang Digunakan Pada *Pseudocode* Praproses dan Segmentasi Pembuluh Darah Retina (Bagian Keenam) 28
Tabel 3. 7 Daftar Fungsi yang Digunakan Pada *Pseudocode* Praproses dan Segmentasi Pembuluh Darah Retina (Bagian Pertama) 28
Tabel 3. 8 Daftar Fungsi yang Digunakan Pada *Pseudocode* Praproses dan Segmentasi Pembuluh Darah Retina (Bagian Kedua) 29
Tabel 3. 9 Daftar Fungsi yang Digunakan Pada *Pseudocode* Praproses dan Segmentasi Pembuluh Darah Retina (Bagian Ketiga) 30
Tabel 3. 10 Daftar Fungsi yang Digunakan Pada *Pseudocode* Praproses dan Segmentasi Pembuluh Darah Retina (Bagian keempat).....31
Tabel 3. 11 Daftar Variabel yang Digunakan Pada *Pseudocode* Ekstraksi Fitur Segmen Vaskular (Bagian Pertama) 40

Tabel 3. 12 Daftar Variabel yang Digunakan Pada *Pseudocode* Ekstraksi Fitur Segmen Vaskular (Bagian Kedua) 41
Tabel 3. 13 Daftar Variabel yang Digunakan Pada *Pseudocode* Ekstraksi Fitur Segmen Vaskular (Bagian Ketiga).....42
Tabel 3. 14 Daftar Fungsi yang Digunakan Pada *Pseudocode* Ekstraksi Fitur Segmen Vaskular (Bagian Pertama) 42
Tabel 3. 15 Daftar Fungsi yang Digunakan Pada *Pseudocode* Ekstraksi Fitur Segmen Vaskular (Bagian Kedua).....43
Tabel 3. 16 Daftar Variabel yang Digunakan Pada *Pseudocode* Klasifikasi Segmen Vaskular (Bagian Pertama).....52
Tabel 3. 23 Daftar Fungsi yang Digunakan Pada *Pseudocode* Klasifikasi Segmen Vaskular (Bagian Pertama).....53

Tabel 5. 1 Tabel Hasil 10 Iterasi.....83
Tabel 5. 2 Hasil klasifikasi berdasar kernel dataset 2:1.....84
Tabel 5. 3 Hasil klasifikasi berdasar kernel dataset 1:1.....84
Tabel 5. 4 *Confusion matrix* Linear-SMO dataset 2:1.....84
Tabel 5. 5 *Confusion matrix* mlp-LS dataset 2:1 85
Tabel 5. 6 *Confusion matrix* quadratic-LS dataset 2:1 86
Tabel 5. 7 *Confusion matrix* polynomial-SMO dataset 2:1 86
Tabel 5. 8 *Confusion matrix* rbf-QP dataset 2:1 87
Tabel 5. 9 *Confusion matrix* linear-LS dataset 1:1 87
Tabel 5. 10 *Confusion matrix* mlp-LS dataset 1:1 88
Tabel 5. 11 *Confusion matrix* quadratic-SMO dataset 1:1 89
Tabel 5. 12 *Confusion matrix* polynomial-SMO dataset 1:1 89
Tabel 5. 13 *Confusion matrix* rbf-SMO dataset 1:1 90
Tabel 5. 14 Hasil klasifikasi akurasi dengan berbagai kernel dataset 1:1 91
Tabel 5. 15 Hasil klasifikasi akurasi dengan berbagai kernel dataset 2:1 92
Tabel 5. 16 *Confusion matrix* polynomial-SMO dataset 1:1 92
Tabel 5. 17 *Confusion matrix* linear-LS dataset 1:1 93
Tabel 5. 18 *Confusion matrix* linear-LS dataset 1:1 94
Tabel 5. 19 *Confusion Matrix* hasil uji coba 94
Tabel 5. 20 Hasil Klasifikasi Tanpa GA dataset 1:1 95

Tabel 5. 21 Hasil Klasifikasi Tanpa GA dataset 2:1 95
Tabel 5. 22 *Confusion matrix* polynomial -QP dataset 1:1 96
Tabel 5. 23 *Confusion matrix* rbf-QP dataset 1:1.....97
Tabel 5. 24 *Confusion matrix* rbf-QP dataset 2:1.....98

[Halaman ini sengaja dikosongkan]

DAFTAR LAMPIRAN

Lampiran A. 1 Dataset Citra Retina (Bagian 1) 103
Lampiran A. 2 Dataset Citra Retina (Bagian 2) 104
Lampiran A. 3 Dataset Citra Retina (Bagian 3) 105
Lampiran A. 4 Dataset Citra Retina (Bagian 4) 106
Lampiran A. 5 Dataset Citra Retina (Bagian 5) 107
Lampiran A. 6 Dataset Citra Retina (Bagian 6) 108
Lampiran A. 7 Dataset Citra Retina (Bagian 7) 109
Lampiran A. 8 Dataset Citra Retina (Bagian 8) 110
Lampiran A. 9 Dataset Citra Retina (Bagian 9) 111
Lampiran A. 10 Dataset Citra Retina (Bagian 10) 112
Lampiran A. 11 Dataset Citra Retina (Bagian 11) 113
Lampiran A. 12 Dataset Citra Retina (Bagian 12) 114
Lampiran A. 13 Data Training dataset 1 (Bagian pertama) 115
Lampiran A. 14 Data Training dataset 1 (Bagian Kedua)..... 116
Lampiran A. 15 Data Test dataset 1 (Bagian Pertama)..... 116
Lampiran A. 16 Data Test dataset 1 (Bagian Kedua)..... 117
Lampiran A. 17 Data Training dataset 2 (bagian pertama)..... 117
Lampiran A. 18 Data Training dataset 2 118
Lampiran A. 19 Data testing dataset 2 119
Lampiran A. 20 Iterasi 20 *run* 1 (5 terbaik) 120
Lampiran A. 21 Iterasi 20 *run* 2 (5 terbaik) 120
Lampiran A. 22 Iterasi 20 *run* 3 (5 terbaik) 120
Lampiran A. 23 Iterasi 20 *run* 4 (5 terbaik) 121
Lampiran A. 24 Iterasi 20 *run* 5 (5 terbaik) 121
Lampiran A. 25 Iterasi 20 *run* 6 (5 terbaik) 121
Lampiran A. 26 Iterasi 20 *run* 7 (5 terbaik) 122
Lampiran A. 27 Iterasi 20 *run* 8 (5 terbaik) 122
Lampiran A. 28 Iterasi 20 *run* 9 (5 terbaik) 122
Lampiran A. 29 Iterasi 20 *run* 10 (5 terbaik) 123
Lampiran A. 30 Iterasi 40 run 1 (5 terbaik)..... 123
Lampiran A. 31 Iterasi 40 run 2 (5 terbaik)..... 123
Lampiran A. 32 Iterasi 40 run 3 (5 terbaik)..... 124
Lampiran A. 33 Iterasi 40 run 4 (5 terbaik)..... 124
Lampiran A. 34 Iterasi 40 run 5 (5 terbaik)..... 124

Lampiran A. 35 Iterasi 40 run 6 (5 terbaik).....125
Lampiran A. 36 Iterasi 40 run 7 (5 terbaik).....125
Lampiran A. 37 Iterasi 40 run 8 (5 terbaik).....125
Lampiran A. 38 Iterasi 40 run 9 (5 terbaik).....126
Lampiran A. 39 Iterasi 40 run 10 (5 terbaik).....126
Lampiran A. 40 Linear-LS dataset 1:1 130
Lampiran A. 41 mlp-SMO dataset 1:1 130
Lampiran A. 42 quadratic-SMO dataset 1:1.....130
Lampiran A. 43 polynomial-SMO dataset 1:1 131
Lampiran A. 44 rbf-SMO dataset 1:1.....131
Lampiran A. 45 linear-SMO dataset 2:1 131
Lampiran A. 46 mlp-LS dataset 2:1 132
Lampiran A. 47 quadratic-LS dataset 2:1.....132
Lampiran A. 48 polynomial-SMO dataset 2:1 132
Lampiran A. 49 rbf-QP dataset 2:1 133
Lampiran A. 50 QP-polynomial dataset 1:1.....133
Lampiran A. 51 LS-rbf dataset 2:1.....133
Lampiran A. 52 QP-polynomial dataset 2:1.....134

DAFTAR KODE SUMBER

Kode Sumber 4. 1 implementasi pemindaian retina.....59
Kode Sumber 4. 2 implementasi *Hessian Matrix*.....60
Kode Sumber 4. 3. implementasi *eigenvalue* 61
Kode Sumber 4. 4 implementasi *Frangi Filter* (bagian pertama)61
Kode Sumber 4. 5. implementasi *Frangi* (bagian Kedua).....62
Kode Sumber 4. 6 implementasi *Frangi* (bagian Ketiga) 63
Kode Sumber 4. 7 implementasi *Mean Filter* 63
Kode Sumber 4. 8 implementasi *Masking* (Bagian Pertama).....63
Kode Sumber 4. 9 implementasi *Masking* (Bagian Kedua) 64
Kode Sumber 4. 10 implementasi *Bradley Thresholding* 64
Kode Sumber 4. 11 implementasi *Opening*.....65
Kode Sumber 4. 12 implementasi *Closing* 65
Kode Sumber 4. 13 implementasi *Program Utama* (Bagian satu) 66
Kode Sumber 4. 14 implementasi *Program Utama* (Bagian kedua) 67
Kode Sumber 4. 15 implementasi Normalisasi Fitur (Bagian Pertama) 67
Kode Sumber 4. 16 implementasi Normalisasi Fitur (Bagian Kedua) 68
Kode Sumber 4. 17 implementasi fitur Area.....68
Kode Sumber 4. 18 implementasi *Fitur Energy* (Bagian Pertama) 68
Kode Sumber 4. 19 implementasi *Fitur Energy* (Bagian Kedua)69
Kode Sumber 4. 20 implementasi *mean gradient* 69
Kode Sumber 4. 21 implementasi *deviasi gradient*.....70
Kode Sumber 4. 22 implementasi *mean gray* 70
Kode Sumber 4. 23 implementasi *Vessel Segment*.....70
Kode Sumber 4. 24 implementasi *Vessel Length* dan *Width Mean* (bagian pertama).....71
Kode Sumber 4. 25 implementasi *crossover*.....72
Kode Sumber 4. 26 Implementasi Mutasi 73

Kode Sumber 4. 27 Implementasi *copychromosome* (Bagian Pertama).....73
Kode Sumber 4. 28 Implementasi *copychromosome* (Bagian Kedua) 74
Kode Sumber 4. 29 implementasi toStr.....74
Kode Sumber 4. 30 Implementasi Generasi Awal(satu) 75
Kode Sumber 4. 31 Implementasi Generasi Awal (dua) 76
Kode Sumber 4. 32 Implementasi program utama (bagian satu) 77
Kode Sumber 4. 33 Implementasi program utama (bagian dua) .78
Kode Sumber 4. 34 Implementasi program utama (bagian tiga) .79

BAB I
PENDAHULUAN

Pada bab ini dibahas hal-hal yang mendasari Tugas Akhir. Bahasan meliputi latar belakang, rumusan masalah, batasan masalah, tujuan, manfaat, metodologi, dan sistematika laporan Tugas Akhir.

1.1. Latar Belakang

Diabetes militus adalah gangguan dari peredaran darah manusia. Diabetes ini disebabkan akan kurangnya insulin di dalam darah. Hal ini dapat menyebabkan meningkatnya tekanan darah yang dapat menyebabkan kerusakan sel pada jaringan tubuh dikarenakan tingginya tekanan darah. Salah satu akibat dari diabetes ini dapat dilihat melalui pembengkakan pembuluh pada jaringan organ [1].

Salah satu jaringan yang dapat terganggu dari pembengkakan pembuluh darah adalah mata. Pada retina mata yang mengalami pembengkakan dapat disebut *proliferative diabetic retinopathy*. Hal ini merupakan kerusakan pada jaringan pembuluh darah retina mata yang menyebabkan cairan keluar dan menciptakan noda pada sekitar jaringan. Noda ini akan menyebabkan terbentuknya jaringan baru (NV) yang dapat mengganggu pengelihatan. Apabila tidak ditangani akan menyebabkan terganggunya pengelihatan penderita hingga dapat berakibat pada kebutaan.

Dalam tugas akhir ini diajukan metode untuk mengambil pola pembuluh darah dari retina yaitu *Bradley Thresholdng*. Sebelumnya citra akan diproses dengan *Hessian matrix, frangi filter* dan *mean filter*. Kemudian akan dilakukan perbaikan dengan morfologi *open* dan *close*.

Pada data yang memiliki fitur yang banyak akan memperberat komputasi. Sebenarnya bisa jadi fitur yang banyak tersebut tidaklah kesemuanya efektif. Sehingga eliminasi fiur yang

dianggap tidak mempengaruhi komputasi perlu dilakukan. Salah satu metode eliminasi fitur yang dapat digunakan adalah Genetic Algorithm (GA). GA menganalogikan fitur sebagai sebuah set kromosom. Kromosom akan mengalami proses mutasi untuk menjadi kromosom yang efektif. Hal ini dapat diperolah dari persilangan antara kromosom yang unggul. Pada GA kromosom yang terbaik akan dijadikan parameter fitur yang baru.

Tugas akhir ini akan mengimplementasikan pengklasifikasian segmen dari vaskular retina mata menggunakan SVM dan seleksi fitur manggunakan GA.

1.2. Rumusan Permasalahan

Rumusan masalah yang diangkat dalam Tugas Akhir ini adalah sebagai berikut:

1. Memahami ekstraksi fiter retina menggunakan pendekatan bentuk fisik.
2. Memahami konsep *Support Vector Machine* sebagai metode pengklasifikasian.
3. Memahami konsep *Genetic Algorithm* sebagai proses seleksi untuk mendapatkan fitur yang efektif.
4. Merancang sistem dan uji coba yang melibatkan penggunaan seleksi fitur *genetic algorithm* dengan SVM.

1.3. Batasan Masalah

Permasalahan yang dibahas dalam Tugas Akhir ini memiliki beberapa batasan, yaitu:

1. Implementasi dari program yang akan dibuat dalam tugas akhir ini menggunakan Matlab.
2. Segmen vaskular retina mata diklasifikasikan ke dalam dua kelas, yaitu vaskular normal dan abnormal.
3. Data yang digunakan untuk data latih dan data uji klasifikasi segmen vaskular berupa citra retina dari basis data STARE (http://www.ces.clemson.edu/~ahoover/stare) [3].

4. Metode Tugas Akhir mengadopsi metode dari *paper* rujukan utama yaitu GA dan SVM [1].
5. Untuk segmentasi menggunakan konfigurasi proses segmentasi terbaik yang telah dilakukan pada tugas akhir milik Friska Ajeng Rizki (5111100135).

1.4. Tujuan dan Manfaat

Tujuan pengerjaan Tugas Akhir ini adalah:

1. Mengetahui konsep dari praproses dan segmentasi untuk menghasilkan citra vaskular retina dengan menerapkan metode yang diajukan.
2. Mengetahui konsep ekstraksi fitur dari citra vascular retina.
3. Mengetahui konsep *Genetic Algorithm* sebagai proses seleksi untuk mendapatkan kombinasi fitur yang efektif.
4. Mengevaluasi kinerja dari segmentasi citra retina serta uji coba pengklasifikasian segmen vaskular retina mata menggunakan Support Vector Machine.

Manfaat dari hasil pembuatan tugas akhir ini adalah menghasilkan program yang dapat membantu ahli dalam menentukan apakah sebuah citra vaskular retina normal atau abnormal. Program ini diharapkan mampu mengurangi kesalahan dari proses klasifikasi sehingga menghasilkan deteksi yang lebih akurat daripada pengklasifikasian secara manual.

1.5. Metodologi

Metodologi yang digunakan pada pengerjaan Tugas Akhir ini adalah sebagai berikut:

1. Penyusunan proposal Tugas Akhir.

Proposal tugas akhir ini berisi tentang deskripsi pendahuluan dari tugas akhir yang akan dibuat. Pendahuluan ini terdiri atas hal yang menjadi latar belakang diajukannya usulan tugas akhir, rumusan masalah yang diangkat, batasan masalah untuk tugas akhir, tujuan dari pembuatan tugas akhir, dan manfaat dari hasil

pembuatan tugas akhir. Selain itu dijabarkan pula tinjauan pustaka yang digunakan sebagai referensi pendukung pembuatan tugas akhir. Sub bab metodologi berisi penjelasan mengenai tahapan penyusunan tugas akhir mulai dari penyusunan proposal hingga penyusunan buku tugas akhir. Terdapat pula sub bab jadwal kegiatan yang menjelaskan jadwal pengerjaan tugas akhir. Akhir dari proposal ini adalah deteksi citra retina mata untuk proliferative diabetic retinopathy menggunakan seleksi fitur genetic algorithm dan berbasis dual classification support vector machine.

2. Studi literatur

Pada studi literatur ini, akan dipelajari sejumlah referensi yang diperlukan dalam pembuatan program yaitu mengenai *frangi's vesselness filter, Bradley thresholding* dan *mean filter*. Hal ini diperlukan untuk mendukung proses praproses dari citra serta klasifikasi. Informasi didapatkan dari sumber penunjang yang berhubungan.

3. Perancangan perangkat lunak

Tahap ini meliputi perancangan sistem berdasarkan studi literatur dan pembelajaran konsep teknologi dari perangkat lunak yang ada. Tahap ini mendefinisikan alur dari implementasi. Langkah-langkah yang dikerjakan juga didefinisikan pada tahap ini. Pada tahapan ini dibuat prototype sistem, yang merupakan rancangan dasar dari sistem yang akan dibuat. Serta dilakukan desain suatu sistem dan desain proses-proses yang ada

4. Implementasi perangkat lunak

Implementasi merupakan tahap membangun rancangan sistem yang telah dibuat. Pada tahapan ini merealisasikan apa yang terdapat pada tahapan sebelumnya, sehingga menjadi sebuah sistem yang sesuai dengan apa yang telah direncanakan.

5. Pengujian dan evaluasi

Pada tahapan ini dilakukan uji coba terhadap perangkat lunak yang telah dibuat. Pengujian dan evaluasi akan dilakukan dengan melihat kesesuaian dengan perencanaan. Tahap ini dimaksudkan juga untuk mengevaluasi jalannya sistem, mencari masalah yang mungkin timbul dan mengadakan perbaikan jika terdapat kesalahan.

6. Penyusunan buku Tugas Akhir.

Pada tahap ini dilakukan penyusunan laporan yang menjelaskan dasar teori dan metode yang digunakan dalam tugas akhir ini serta hasil dari implementasi aplikasi perangkat lunak yang telah dibuat.

1.6. Sistematika Laporan

Buku Tugas Akhir ini disusun dengan sistematika sebagai berikut:

BAB I. PENDAHULUAN

Bab yang berisi mengenai latar belakang, tujuan, dan manfaat dari pembuatan Tugas Akhir. Selain itu permasalahan, batasan masalah, metodologi yang digunakan, dan sistematika penulisan juga merupakan bagian dari bab ini.

BAB II. DASAR TEORI

Bab ini berisi penjelasan secara detail mengenai dasar-dasar penunjang dan teori-teori yang digunakan untuk mendukung pembuatan Tugas Akhir ini.

BAB III. PERANCANGAN PERANGKAT LUNAK

Bab ini berisi tentang desain sistem yang disajikan dalam bentuk *pseudocode*.

BAB IV. IMPLEMENTASI

Bab ini membahas implementasi dari desain yang telah dibuat pada bab sebelumnya. Penjelasan berupa *code* yang digunakan untuk proses implementasi.

BAB V. UJI COBA DAN EVALUASI

Bab ini menjelaskan kemampuan perangkat lunak dengan melakukan pengujian kebenaran dan pengujian kinerja dari sistem yang telah dibuat.

BAB VI. KESIMPULAN DAN SARAN

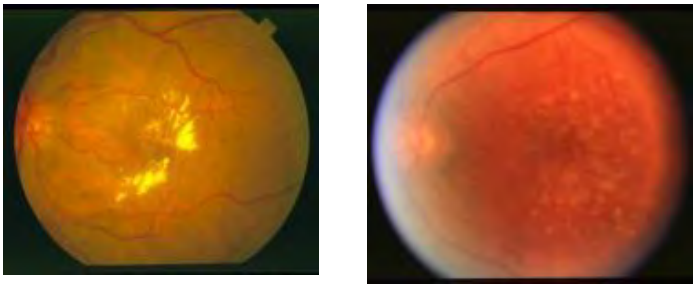
Bab ini merupakan bab terakhir yang menyampaikan kesimpulan dari hasil uji coba yang dilakukan dan saran untuk pengembangan perangkat lunak ke depannya.

BAB II
TINJAUAN PUSTAKA

Bab ini berisi penjelasan teori-teori yang berkaitan dengan algoritma yang diajukan pada pengimplementasian perangkat lunak. Penjelasan ini bertujuan untuk memberikan gambaran secara umum terhadap sistem yang dibuat dan berguna sebagai penunjang dalam pengembangan perangkat lunak.

2.1. Praproses Citra Retina

Pada kasus *proliverative diabetic retinopathy* dapat dikenali dengan menganalisa citra retina. Hal ini dapat dilakukan dengan mengamati pembuluh darah yang ada pada citra retina. Pembuluh darah manyimpan informasi PDR karena terjadi perubahan struktur pada retina karena terdapat *New Vessel*. Namun tidak semua citra retina dapat dipakai secara langsung dikarenakan kualitas citra yang kurang baik sehingga antara latar dengan jaringan pembuluh darah kurang bisa dibedakan seperti Gambar 2.1.



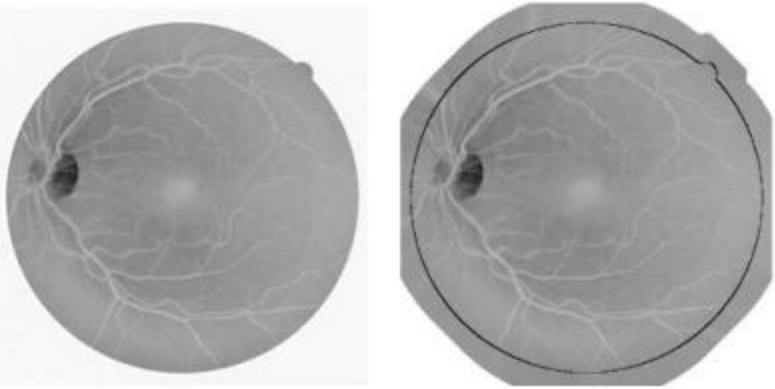
Gambar 2. 1 Citra retina dari basis data STARE [3]

Untuk mengatasi permasalahan tersebut dilakukanlah praproses citra. Praproses yang dilakukan mencakup dua hal yaitu *masking* dan pengambilan citra menggunakan *channel* warna tertentu dalam hal ini *green channel*.

2.1.1 *Masking*

Masking adalah pemisahan objek dengan latar belakang. Dalam hal ini *masking* digunakan untuk memisahkan objek retina dengan latar belakang *masking*. Hal ini digunakan karena dalam dataset yang digunakan, *background* citra tidak benar-benar hitam. Hal ini akan menggangu proses segmentasi apabila dibiarkan.

Masking adalah sebuah *thresholding* sederhana dengan menetapkan suatu nilai sebagai nilai ambang batas atau *threshold*. Sebagai masukan dari proses *masking* dibutuhkan input *greyscale* dari citra retina dengan skala 0-255. Citra dalam *inverse green channel* dan citra hasil praproses ditunjukkan pada Gambar 2.2 [4].

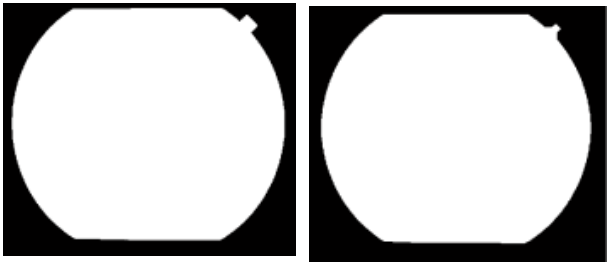


Gambar 2. 2 Citra retina dalam *inverse green channel* dan citra hasil praproses

2.1.2 *Green Channel*

Pada citra retina asli terkadang sulit untuk membedakan antara pembuluh darah dan bukan. Hal ini dapat mengurangi akurasi dari hasil segmentasi. Pada citra asli retina merupakan gabungan antara beberapa warna yaitu *red, green,* dan *blue* atau biasa disebut RGB. Dalam beberapa referensi paper menunjukkan bahwa penggunaan *channel* warna tertentu dapat meningkatkan

hasil segmentasi. Menurut penggunaan *green channel* dapat meningkatkan hasil segmentasi dikarenakan kontras antara pembuluh darah dengan latar paling bagus diantara channel lainnya [2]. Hasil *masking* menggunakan kanal hijau dapat dilihat pada gambar 2.3.



Gambar 2. 3 Citra *masking* dari *Green Channel*

2.2. Segmentasi Vaskular Retina

Proses segmentasi vaskular retina adalah proses yang bertujuan mengambil pola garis pembuluh darah dari citra retina. Hal ini diperlukan karena citra pembuluh darah dapat dijadikan objek pengamatan dalam mendeteksi *proliferative diabetic retinopathy*. Dalam tugas akhir ini diajukan metode untuk mengambil pola pembuluh darah dari retina yaitu *Bradley Thresholdng*. Sebelumnya citra akan diproses dengan *Hessian matrix, frangi filter* dan *mean filter*. Kemudian akan dilakukan perbaikan dengan morfologi *open* dan *close*.

2.2.1 *Hessian Matrix*

Hessian matrix adalah algoritma optimasi yang menerapkan matrik persegi dengan setiap elemen pembentuknya adalah turunan orde ke dua dari fungsi sebuah citra. Persamaan hessian matrik dapat didefinisikan dengan rumus 2.1 dan 2.2.

$$H = \begin{bmatrix} f_{xx} & f_{xy} \\ f_{yx} & f_{yy} \end{bmatrix} \quad (2.1)$$

$$f_{xx} = \frac{\partial^2 f}{\partial x^2}, \quad f_{xy} = \frac{\partial^2 f}{\partial xy}, \quad f_{yx} = \frac{\partial^2 f}{\partial yx}, \quad f_{yy} = \frac{\partial^2 f}{\partial y^{2\prime}} \quad (2.2)$$

dimana f_{xx}, f_{xy}, f_{yx}, dan f_{yy} adalah deviasi orde kedua dari f pada *Hessian matrix*.

2.2.2 *Eigenvalue* dan Vektor

Eigenvalue adalah sebuah nilai karakteristik matrix yang memiliki persamaan pada persamaan linear. Persamaan *eigenvalue* seperti 2.3.

$$A\,x = \lambda x \quad (2.3)$$

Dimana A adalah matriks berukuran n*n, x adalah vector dan λ adalah *eigenvalue*. Cara perhitungan untuk mendapatkan λ didefinisikan dengan persamaan 2.4.

$$\lambda_1 = K - \sqrt{K^2 - Q^2}, \quad \lambda_2 = K - \sqrt{K^2 - Q^2} \quad (2.4)$$

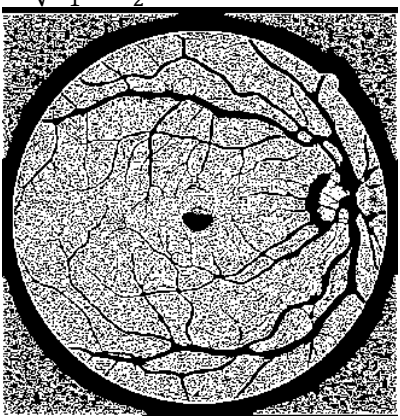
dimana $K = (f_{xx} + f_{yy})/2$ serta $Q = \sqrt{f_{xx}f_{yy} + f_{xy}f_{yx}}$ dengan f_{xx}, f_{xy}, f_{yx}, dan f_{yy} yang merupakan nilai dari *Hessian matrix* pada Persamaan 2.1.

2.2.3 *Frangi Filter*

Frangi filter berfungsi sebagai perbaikan kualitas gambar dengan cara memperjelas bagian yang akan diukir serta memperkirakan radius pembuluh. Selain itu akan membuat kontras antara latar dan objek semakin tinggi. Pengukuran terhadap citra pembuluh dapat dilakukan dengan melakukan *sorting* $\lambda_1 > \lambda_2$ dan *postulating* $\lambda_1 = \lambda_2 < 0$. Untuk melakukan proses pengukuran citra pembuluh ini, dapat menggunakan fungsi dari Persamaan 2.5 [5].

$$v(\sigma) = \begin{cases} 0 & if\ \lambda_1 > 0 \\ \exp\left(-\frac{R_B^2}{2\beta^2}\right)(1 - \exp(\frac{-S^2}{2c^2})) & otherwise \end{cases} \quad (2.5)$$

dimana σ merupakan standar deviasi pada skala spasial, serta β dan c masing-masing merupakan nilai konstanta koreksi. Untuk dapat menghasilkan citra yang baik, pada β diberikan nilai sebesar 0.5 dan c diberikan nilai sebesar 15. Selain itu, juga terdapat $R_B = \frac{\lambda_2}{\lambda_1}$, dan $S = ||H||F = \sqrt{\lambda_1^2 + \lambda_2^2}$.



Gambar 2. 4 Citra hasil dari *Frangi filter*

2.2.4 *Mean Filter*

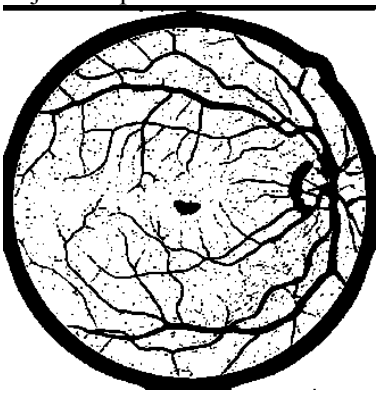
Mean filter merupakan *filter* spasial geser-jendela (*sliding-window*) yang menggantikan nilai tengah dari jendela dengan rata-rata dari semual nilai piksel dari jendela dengan ukuran N x N. Ukuran N pada jendela dapat disesuaikan tergantung pada kebutuhan. Secara umum, persamaan dari *mean filter* dapat dilihat pada Persamaan 2.6 [6].

$$h[i,j] = \frac{1}{M}\sum_{(k,l)\in N} f[k,l] \qquad (2.6)$$

dimana M merupakan total jumlah piksel (hasil perkalian) dari jendela N [7].

Mean filter digunakan dengan tujuan untuk mereduksi jumlah intensitas variasi antara satu piksel dengan piksel lainnya [6]. Adapun citra hasil dari proses *mean filter* menggunakan citra

hasil proses deteksi sebagai masukan dan nilai jendela dengan ukuran 3 x 3 yang ditunjukkan pada Gambar 2.3.



Gambar 2. 5 hasil *mean filter*

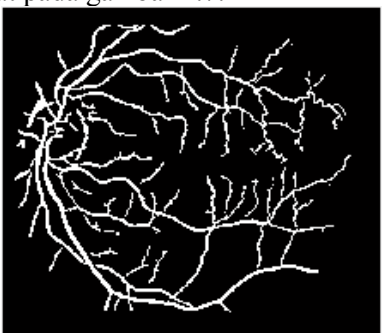
2.2.5 *Bradley Local Image Thresholding*

Bradley threshold adalah sebual algoritma thresholding yang bersifat adaptive. Pada sebuah citra, algoritma ini akan mengatur setiap piksel retina pada nilai warna hitam apabila dia memiliki tingkat kecerahan lebih rendah dari T persen begitupun sebaliknya. Berikut hasil gambar dari *Bradley thresholding* [8].

Gambar 2. 6 hasil *Bradley Threshold*

2.2.6 *Morphological Cleaning* dan *Image Enhancement*

Proses morfologi mencakup dua hal utama yaitu proses *closing* dan *opening.* Pada *opening* akan dilakukan pembersihan citra hasil thresholding dengan cara menghapus area objek yang kurang dari sebuah nilai *opening*. Sedangkan pada proses *closing* akan dilakukan penambahan area pada lubang suatu objek. Kedua proses tersebut dapat dikatakan sebagai proses pembersihan pada proses *opening* dan perbaikan kualitas citra pada *closing* [9]. Hasil akhir dapat dilihat pada gambar 2.7.



Gambar 2. 7 Citra hasil akhir segmentasi

2.3. Ekstraksi Fitur Vaskular Retina

Untuk mendapatkan sebuah kelas dari sebuah citra diperlukanlah penggalian informasi dari sebuah citra. Hal ini disebut proses ekstraksi fitur. Ekstraksi fitur sendiri merupakan langkah penting sebelum kita melangkah pada proses klasifikasi. Dalam proses klasifikasi sendiri diperlukan sebuah inputan berupa *feature vector* yang merupakan kumpulan fitur atau ciri yang diekstraksi dari sebuah citra. *Feature vector* adalah sebuah set yang berisi beberapa nilai hasil ekstraksi fitur tertentu.

1.3.1 *Pixel Area*

Area yaitu jumlah piksel yang merupakan pembuluh darah di dalam jendela. Nilai *area* diperoleh dengan menghitung piksel yang bernilai 1, karena pada citra biner hasil segmentasi, pembuluh darah digambarkan dengan piksel berwarna putih [1].

1.3.2 *Energy*

Energy yaitu intensitas dari piksel pembuluh darah pada segmen kandidat. Nilai *energy* didapatkan dengan menghitung jumlah intensitas *green channel* dari semua piksel yang merupakan pembuluh darah. Pembuluh darah normal cenderung lebih terang dibandingkan dengan pembuluh darah abnormal pada *inverse green channel* citra [1].

1.3.3 *Mean Gradient*

Mean gradient adalah nilai rataan *gradient magnitude* dari piksel pembuluh darah pada segmen kandidat. *Gradient* dari sebuah citra dengan menghitung perubahan intensitas dari setiap titik pada citra *gradient* terhadap titik yang sama pada citra asli dalam arah tertentu. *Gradient* dari citra memiliki dua komponen arah, yaitu (*x,y*) dimana fungsi *gradient* dari (*x,y*) adalah turunan parsial pada arah *x* dan *y*. *Gradient magnitude* didefinisikan sebagai $\nabla f = \frac{\partial f}{\partial x}\hat{x} + \frac{\partial f}{\partial y}\hat{y}$ dengan $\frac{\partial f}{\partial x}$ adalah *gradient* pada arah *x* dan $\frac{\partial f}{\partial y}$ adalah *gradient* pada arah *y*.

Dalam kasus ini, nilai *green channel* dari citra retina yang akan dihitung dengan *gradient magnitude*. *Mean gradient* dihitung dengan menjumlahkan semua nilai *gradient magnitude* pada setiap piksel kemudian dibagi dengan luas jendela [1].

1.3.4 *Standard Deviation Gradient*

Standard deviation gradient adalah standar deviasi dari *gradient magnitude* dari piksel pembuluh darah pada segmen kandidat. *Standard deviation gradient* merupakan akar kuadrat dari hasil penjumlahan kuadrat selisih *mean gradient* dengan nilai *gradient magnitude* tiap piksel dibagi dengan luas jendela [1].

1.3.5 *Mean Grayscale*

Mean Grayscale adalah nilai rata-rata skala keabuan dari seluruh pixel dalam sebuah citra. Sebelumnya dari citra asli diambil nilai intensitas dalam skala keabuan pada *green channel*. Kemudian dihitung jumlah nilai *gray scale* pada setiap piksel. Hasil penjumlahan tersebut kemudian dicari *mean* dengan membagi jumlah nilai penjumlahan dari tiap piksel dengan luas jendela citra [1].

1.3.6 *Vessel Segment*

Vessel segment adalah jumlah segmen pembuluh darah pada kandidat segmen. Jumlah segmen ditentukan dengan menggunakan *connected component analysis*. *Connected component* bekerja dengan memeriksa apakah piksel tetangga memiliki intensitas yang sama dengan piksel tersebut. Jika piksel tetangga memiliki intensitas yang sama, maka piksel tetangga tersebut akan dilabeli sebagai piksel yang terdapat pada *region* dengan piksel tersebut. Pembuluh darah abnormal biasanya memiliki jumlah segmen yang lebih banyak dibandingkan pembuluh darah normal [1].

1.3.7 *Vessel Width Mean*

Vessel width mean adalah fitur yang berisikan nilai rata-rata dari lebar rata-rata dari sebuah segmen yang ada dalam sebuah citra retina. Nilai ini diperoleh dengan memberikan asumsi bahwa

apabila sebuah *vessel* ditarik lurus akan berbentuk persegi panjang. Hal pertama yang dilakukan adalah melabeli tiap vessel kemudian menghitung anggota piksel bernilai 1 dari setiap label kemudian hasil tersebut dibagi dengan panjang tiap vessel [1].

1.3.8 *Vessel Length Mean*

Vessel length mean adalah sebuah fitur yang berisikan nilai panjang rata-rata semua segmen dalam sebuah citra. Hal ini dapat diperoleh dengan melakukan proses *thinning* pada sebuah citra sehingga citra hanya memiliki lebar sebesar 1 piksel kemudian dilakukan pelabelan dan dihitung jumlah anggota piksel bernilai 1 untuk setiap label [1].

2.4. Seleksi Fitur *Genetic Algorithm*

Genetic Algorithm adalah sebuah proses untuk memilih komposisi fitur terbaik dari sebuah set fitur. Algoritma ini akan menyeleksi fitur dengan sebuah fungsi objektif tertentu yang biasa disebut *fitness function*. Fungsi tersebut dapat berupa nilai akurasi ataupun running time terbaik. Dalam tugas ini *fitness function* yang dipakai adalah dengan akurasi terbaik yang diperoleh dari *testing* data.

2.4.1 Proses Utama *Genetic Algorithm*

Genetic Algorithm adalah sebuah algoritma seleksi fitur dengan pemodelan *chromosome* yang berupa *feature vector* [4]. Dalam hal ini *genetic algorithm* mengadaptasi proses seleksi alam dimana hanya kromosom terbaiklah yang akan dibawa pada generasi berikutnya. Berikut adalah proses utama yang ada dalam *Genetic Algorithm*:

a. Pembentukan Generasi Awal Kromosom

Generasi awal ini adalah kromosom yang dibuat sejumlah nilai tertentu yang bertujuan sebagai objek untuk dilakukan *genetic algorithm.*

b. Desain Kromosom

Desain kromosom yang digunakan adalah *binary* kromosom. Kromosom ini berisi nilai 0 atau 1 sebanyak fitur yang kemudian merepresentasikan bahwa fitur ke i tersebut tarpakai apabila kromosom ke i bernilai 1.

c. *Crossover*

Crossover adalah sebuah proses persilangan antara 2 atau lebih kromosom. Proses persilangan ini dilakukan dengan menggabungkan sebagian kromosom *i* dengan sebagian kromosom *j*. Pemilihan kromosom ini menggunakan sebuah nilai tertentu yang disebut nilai *threshold.* Nilai tersebut adalah nilai dari *fitness function* dari kromosom tersebut. Proses persilangan yang digunakan dalam tugas akhir ini adalah jenis *random one point crossover*. Persilangan jenis ini akan merandom sebuah titik diantara 2 hingga n-1 dimana n adalah jumlah fitur sehingga hasil nilai random tersebut dijadikan sebuah acuan *crossover* kromosom.

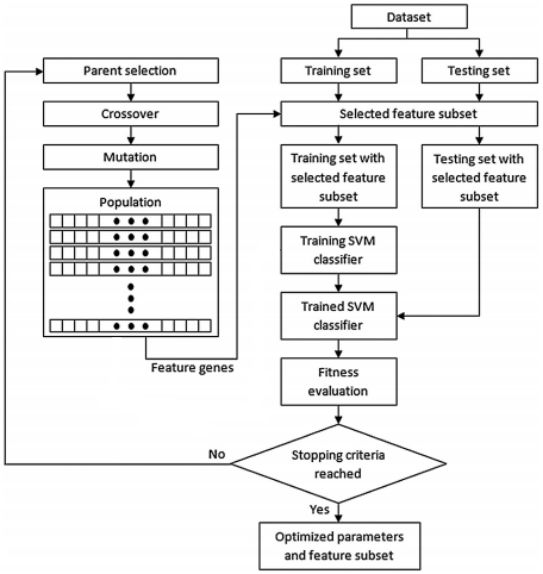
d. *Mutation*

Mutation adalah proses perubahan susunan dari sebuah kromosom. Perubahan ini berupa *inverse* dari gen kromosom dengan indeks tertentu. Mutasi memiliki laju mutasi dimana merupakan nilai probabilitas sebuah gen dalam kromosom mengalami mutasi. Pada tugas akhir ini jenis mutasi yang digunakan adalah *random point mutation*. Metode ini akan merandom point dari 1 hingga sebanyak jumlah fitur sebanyak *n* kali sesuai laju mutasi sebuah kromosom.

Secara umum urutan proses dari GA adalah dimulai dari pembentukan awal kromosom. Hal ini dilakukan agar ada kromosom awal yang dapat dijadikan acuan untuk proses selanjutnya. Kromosom awal ini akan terlebih dahulu dinilai *fitness function*-nya untuk kemudian dilakukan sorting. Kemudian diambillah beberapa kromsom dari kromosom yang berada pada *rank* atas untuk dilakukan *crossover. Crossover* akan menyilangkan kromosom satu dengan yang lainnya untuk kemudian dibawa ke proses mutasi. Dalam mutasi kromosom akan

mengalami perubahan kode genetik sesuai dengan laju mutasi. Kromosom hasil mutasi merupakan kromosom baru yang kemudian dievaluasi berdasarkan *fitness function*-nya. Hal ini berlanjut hingga proses memenuhi *stopping criteria*.

Untuk detail cara kerja *genetic algorithm* dapat dilihat pada gambar 2.8.



Gambar 2. 8 Alur GA dan SVM [1]

2.4.2 *Fitness Function*

Fitness function adalah sebuah nilai dari sebuah kromosom. Nilai ini didapatkan dengan penghitungan sebuah aspek yang dapat dihasilkan dari kromosom tersebut. Salah satu aspek yang dapat diambil dari sebuah kromosom adalah nilai akurasi yang dihasikan sebuah set fitur yang berada dalam kromosom tersebut karena objektif dari proses *genetic algorithm* pada tugas akhir ini adlah untuk mencari kromosom dengan kombinasi set fitur yang

menghasilkan nilai akurasi terbaik. Fungsi fitness didefinisikan dengan persamaaan 2.7 [1].

$$error = \frac{FP+FN}{TP+FP+FN+TN} \quad (2.7)$$

2.5. *Support Vector Machine*

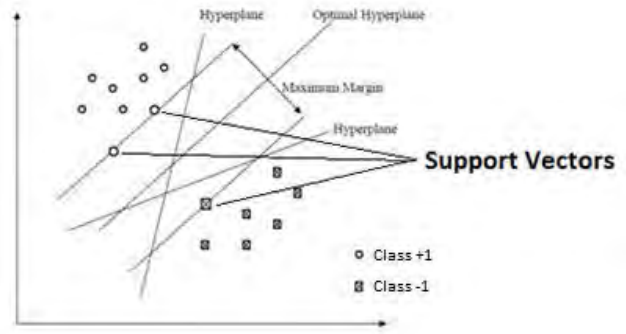
Konsep dasar SVM secara sederhana adalah usaha mencari *hyperplane* terbaik yang berfungsi sebagai pemisah dua kelas pada *input space*. *Hyperplane* atau disebut *support vector* adalah sebuah nilai yang digunakan sebagai pembobotan nilai fitur untuk mencapai sebuah fungsi objektif tertentu. Hal ini didapat dari titik-titik yang berada di dekat garis *hyperplane* data inilah yang disebut *support vector*. Secara matematis SVM dapat dirumuskan seperti persamaan [10].

$$w \,.\, x + b = 0 \quad (2.8)$$

$$w \,.\, x_i + b \leq 0 \;\; then - 1 \quad (2.9)$$

$$w \,.\, x_i + b \geq 0 \;\; then + 1 \quad (2.10)$$

Dimana *w* adalah *hyperplane*, *x* adalah fitur, dan *b* adalah suatu nilai bias tertentu. Pada SVM *w* adalah *support vector* yang akan membuat *hyperplane* seperti gambar 2.9. SVM pada dasarnya memisahkan kelas melalui batas 0, sehingga apabila nilai persamaan 2.8 kurang dari 0 maka kelas -1 dan sebaliknya maka kelas -1 sesuai persamaan 2.10.



Gambar 2. 9 Ilustrasi *hyperplane* SVM

2.6. *Kernel Trick*

Apabila sebuah data memiliki karakteristik *linearly separable* maka data tersebut aan mudah dipetakan oleh SVM. Namun apabila data *non linearly separable* maka diperlukan teknik tertentu agar solusi dari SVM dapat maksimal. Teknik yang digunakan biasa disebut *kernel trick* [1].

Fungsi kernel akan memetakan *x* dengan suatu fungsi *f(x)* sehingga *x'* adalah representasi x dalam suatu ruang vektor tertentu. Ruang vektor ini adalah ruang vektor yang dimana *hyperplane* dapat dibentuk. Untuk persamaan kernel dapat dilihat pada persamaan 2.11.

$$\mathrm{K}(x_i, x_j) = \Phi(x_i).\Phi(x_j) \qquad (2.11)$$

Kernel trick memberikan berbagai kemudahan, karena dalam proses pembelajaran SVM, untuk menentukan *support vector*, kita hanya cukup mengtahui fungsi *kernel* yang dipakai, dan tidak perlu mengetahui wujud dari fungsi *non linear* Φ. Jenis-jenis *kernel* yang umum digunakan dalam SVM dapat dilihat pada tabel 2.1.

Tabel 2. 1 Trik Kernel SVM yang umum

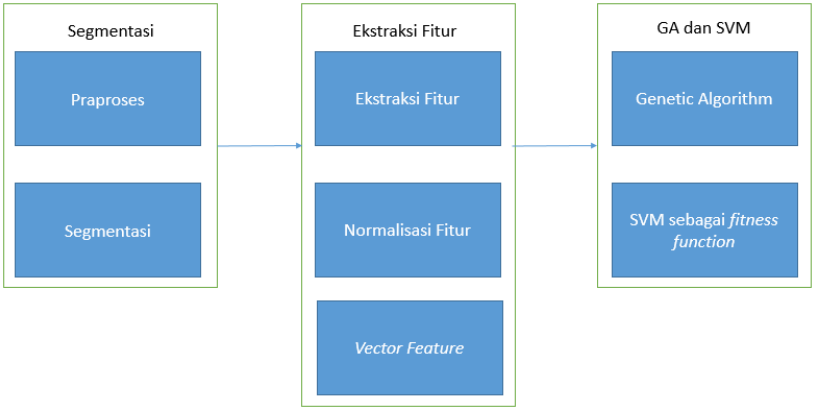
Jenis kernel	Definisi
Polynomial	$\mathrm{K}(x_i, x_j) = (x_i.x_j + 1)^p$
Gaussian	$\mathrm{K}(x_i, x_j) = \exp(\frac{\lvert\lvert x_i - x_j \rvert\rvert^2}{2\sigma^2})$

BAB III
PERANCANGAN

Pada bab ini akan dijelaskan perancangan perangkat lunak pengklasifikasi citra retina. Perancangan akan dibagi menjadi tiga proses utama yaitu

1. praproses citra dan segmentasi vaskular retina,
2. proses ekstraksi fitur dari segmen vaskular retina,
3. serta klasifikasi yang dipadukan dengan *genetic algorithm.* serta pengujian melalui *support vector machine*.

Untuk masing-masing proses utama akan dibagi menjadi proses-proses kecil yang terlibat didalamnya. Untuk gambaran umum dapat dilihat pada gambar 3.1. Sebelumnya pada bab ini akan dijelaskan gambaran umum setiap program utama dalam bagan gambaran umum metode, selanjutnya untuk penjelasan lebih detail akan disajikan dalam *pseudocode.*



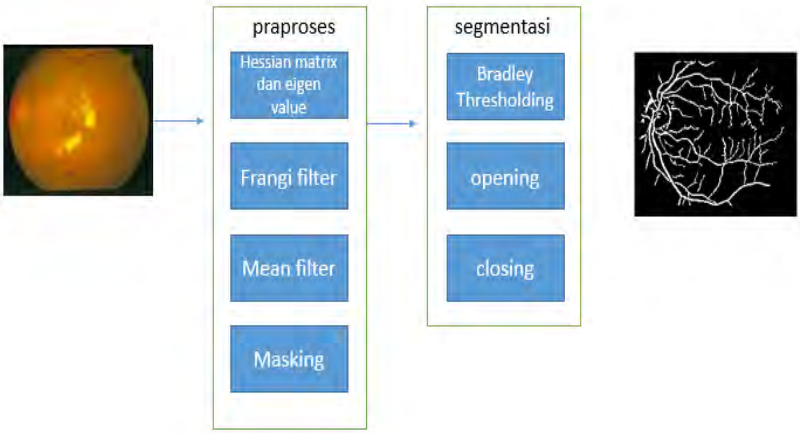
Gambar 3. 1 Gambaran umum proses keseluruhan

3.1. Perancangan Praproses dan Segmentasi Vaskular

Dalam tahap ini diawali dengan tahapan praproses. Tahap ini citra asli retina akan diolah dengan *hessian matrix* untuk mendapatkan *eigenvalue*. Setelah itu dilakukan pengukuran

pembuluh darah dengan *frangi filter* dan *mean filter*. Setelah hasil dari *mean filter* akan dipisahkan *foreground dan background* melalui proses *masking.* Hasil dari masking merupakan inputan untuk proses segmentasi. Tahap ini diawali dengan *Bradlay thresholding* untuk kemudian diperbaiki melalui proses *opening* dan *closing*. Untuk gambaran umum proses dapat dilihat pada gambar 3.1.

Pseudocode proses utama ini terdiri dari satu program utama yang memliki tujuan untuk memanggil beberapa program kecil lainnya yang berisi proses-proses yang lebih kecil untuk dijalankan. Pada gambar 3.1 dapat dilihat proses utama praproses dan segmentasi. Dapat dilihat bahwa proses ini diawali dengan menggunakan masukan berupa citra awal retina yang masih berupa RGB. Citra tersebut kemudian akan diolah dalam praproses terlebih dahulu dengan menggunakan metode hessian matrik dan eigen value. Kemudian hasilnya akan diproses dengan frangi filter, mean filter serta masking. Citra hasil masking akan digunakan untuk segmentasi menggunakan bradley *thresholding*. Hasil dari thresholding kemudian akan diperbaiki melalui morfologi *open* dan *close*.



Gambar 3. 2 Proses utama praproses dan segmentasi

Tabel 3. 1 Daftar Variabel yang Digunakan Pada *Pseudocode* Praproses dan Segmentasi Pembuluh Darah Retina (Bagian Pertama)

No.	Nama Variabel	Tipe	Penjelasan
1.	`img`	uint8	Citra asli retina dalam ruang warna RGB dengan ukuran 584 x 565 piksel.
2.	`Im`	double	Citra asli retina yang telah diubah ke dalam tipe *double* dengan ukuran 584 x 565 piksel.
3.	`input`	double	Citra asli yang telah diubah ke dalam ruang warna hijau (kanal hijau) dengan ukuran 584 x 565 piksel.
4.	`options`	double	Berupa *struct* berisi atribut nilai *default* yang diberlakukan untuk dapat menjalankan fungsi *Frangi filter*.
5.	`frangiv`	double	Citra biner yang merupakan hasil dari proses deteksi (*Frangi filter*) dengan ukuran 584 x 565 piksel.
6.	`sigmas`	double	Nilai skala yang digunakan untuk mengatur *range* dari sigma yang digunakan pada *Frangi*.
7.	`beta`	double	Nilai konstanta koreksi β pada fungsi *Frangi filter* yang telah dikuadratkan dan dikalikan dua.
8.	`c`	double	Nilai konstanta koreksi c pada fungsi *Frangi filter* yang telah dikuadratkan dan dikalikan dua.
9.	`ALLfiltered`	double	Matriks yang digunakan sebagai penyimpanan citra hasil *filter* untuk Ifiltered.

Tabel 3. 2 Daftar Variabel yang Digunakan Pada *Pseudocode* Praproses dan Segmentasi Pembuluh Darah Retina (Bagian Kedua)

No.	Nama Variabel	Tipe	Penjelasan
8.	`c`	double	Nilai konstanta koreksi *c* pada fungsi *Frangi filter* yang telah dikuadratkan dan dikalikan dua.
9.	`ALLfiltered`	double	Matriks yang digunakan sebagai penyimpanan citra hasil *filter* untuk Ifiltered.
10.	`ALLangles`	double	Matriks yang digunakan sebagai penyimpanan citra hasil *filter* untuk angles.
11.	`Dxx`	double	Berupa nilai deviasi kedua pada xx dari *Hessian matrix* yang digunakan untuk mendapatkan *eigenvalue*.
12.	`Dxy`	double	Berupa nilai deviasi kedua pada xy dari *Hessian matrix* yang digunakan untuk mendapatkan *eigenvalue*.
13.	`Dyy`	double	Berupa nilai deviasi kedua pada yy dari *Hessian matrix* yang digunakan untuk mendapatkan *eigenvalue*.
14.	`angles`	double	Berupa nilai hasil komputasi direksi dari vektor eigen minor.
15.	`Rb`	double	Nilai maksimum dari hasil kuadrat yang dilakukan operasi pembagian antar *eigenvalue*.
16.	`S2`	double	Nilai maksimum dari hasil penjumlahan antar *eigenvalue* yang telah dikuadratkan.
17.	`Ifiltered`	double	Nilai maksimum dari hasil pengukuran kemiripan dari citra.

Tabel 3. 3 Daftar Variabel yang Digunakan Pada *Pseudocode* Praproses dan Segmentasi Pembuluh Darah Retina (Bagian Ketiga)

No.	Nama Variabel	Tipe	Penjelasan
18.	`outIm`	double	Berupa *array* yang menyimpan nilai piksel hasil proses *Frangi filter*.
19.	`whatScale`	double	Berupa *array* yang menyimpan nilai skala hasil proses *Frangi filter*.
20.	`Direction`	double	Berupa *array* yang menyimpan nilai direksi hasil proses *Frangi filter*.
21.	`DGaussxx`	double	Berupa *filter* yang digunakan untuk menghitung nilai deviasi kedua pada xx.
22.	`DGaussxy`	double	Berupa *filter* yang digunakan untuk menghitung nilai deviasi kedua pada xy.
23.	`DGaussyy`	double	Berupa *filter* yang digunakan untuk menghitung nilai deviasi kedua pada yy.
24.	`tmp`	double	Nilai untuk menghitung nilai vektor eigen J.
25.	`v2x`	double	Nilai untuk menghitung nilai vektor eigen v1.
26.	`v2y`	double	Nilai untuk menghitung nilai vektor eigen v2.
27.	`mag`	double	Normalisasi dari nilai vektor eigen pada v2x dan v2y.
28.	`v1x`	double	Variabel sementara untuk menyimpan nilai ortogonal dari vektor eigen v2x.

Tabel 3. 4 Daftar Variabel yang Digunakan Pada *Pseudocode* Praproses dan Segmentasi Pembuluh Darah Retina (Bagian Pertama)

No.	Nama Variabel	Tipe	Penjelasan
29.	`v1y`	double	Variabel sementara untuk menyimpan nilai ortogonal dari vektor eigen v2y.
30.	`mu1`	double	Variabel sementara untuk menyimpan *eigenvalue* dari λ_2.
31.	`mu2`	double	Variabel sementara untuk menyimpan *eigenvalue* dari λ_1.
32.	`check`	double	Variabel untuk mengurutkan *eigenvalue* dengan $\lambda_1 < \lambda_2$.
33.	`Lambda1`	double	Berupa *eigenvalue* dari λ_2.
34.	`Lambda2`	double	Berupa *eigenvalue* dari λ_1.
35.	`Ix`	double	Nilai ortogonal dari vektor eigen v2x.
36.	`Iy`	double	Nilai ortogonal dari vektor eigen v2y.
37.	`img2`	uint8	Citra biner hasil akhir dari proses deteksi pembuluh darah retina dengan ukuran 584 x 565 piksel.
38.	`mean`	double	Variabel sementara yang digunakan untuk menyimpan nilai dari fungsi *mean filter*.
39.	`meanf`	uint8	Citra biner hasil proses *mean filter* dengan ukuran 584 x 565 piksel.
40.	`circleImage`	logical	Citra *mask* yang akan digunakan untuk melakukan proses *masking*.
41.	`rows`	double	Nilai *rows* (baris) yang didapatkan dari citra.
42.	`columns`	double	Nilai *columns* (kolom) yang didapatkan dari citra.

Tabel 3. 5 Daftar Variabel yang Digunakan Pada *Pseudocode* Praproses dan Segmentasi Pembuluh Darah Retina (Bagian Kelima)

No.	Nama Variabel	Tipe	Penjelasan
43.	`numberOfColorBands`	double	Nilai warna yang didapatkan dari citra.
44.	`img3`	uint8	Citra biner pembuluh darah yang digunakan untuk proses *masking* dengan ukuran 584 x 565 piksel.
45.	`mask`	uint8	Citra *mask* dengan ukuran 584 x 565 piksel.
46.	`maskedImage`	uint8	Citra biner hasil dari proses *masking* dengan ukuran 584 x 565 piksel.
47.	`bt`	logical	Citra biner hasil dari proses *Bradley thresholding* dengan ukuran 584 x 565 piksel.
48.	`image`	uint8	Citra masukan yang digunakan untuk melakukan proses suatu fungsi.
49.	`varargin`	cell	Variabel masukan dalam suatu fungsi yang digunakan untuk menerima sejumlah argumen masukan.
50.	`numvarargs`	int	Jumlah panjang (*length*) variabel yang didapatkan dari varargin.
51.	`optargs`	cell	Nilai yang akan digunakan pada fungsi apabila nilai argumen yang dimasukan pada fungsi tersebut tidak memiliki jumlah panjang (*length*) yang sama.
52.	`window`	int	Nilai piksel jendela atau matriks ketetanggaan (*neighbourhood*) dari suatu fungsi.

Tabel 3. 6 Daftar Variabel yang Digunakan Pada *Pseudocode* Praproses dan Segmentasi Pembuluh Darah Retina (Bagian Keenam)

No.	Nama Variabel	Tipe	Penjelasan
53.	`T`	int	Nilai persentase tingkat kecerahan pada citra yang akan dilakukan proses fungsi bradley.
54.	`padding`	char	Menunjukkan *padding* yang akan digunakan untuk *filter* pada suatu fungsi.
55.	`avrg`	double	Variabel sementara yang digunakan untuk menyimpan nilai dari fungsi *average filter*.
56.	`output`	uint8	Citra keluaran yang dihasilkan dari proses suatu fungsi.
57.	`mo`	logical	Citra biner hasil dari proses morfologi *area opening* dengan ukuran 584 x 565 piksel.
58.	`se`	strel	Variabel sementara yang digunakan untuk menyimpan nilai dari fungsi *structuring element*.
59.	`mc`	logical	Citra biner hasil dari proses morfologi *close* dengan ukuran 584 x 565 piksel.

Tabel 3. 7 Daftar Fungsi yang Digunakan Pada *Pseudocode* Praproses dan Segmentasi Pembuluh Darah Retina (Bagian Pertama)

No.	Nama Fungsi	Penjelasan
1.	`double`	Fungsi untuk mengubah suatu nilai ke dalam presisi *double*.
2.	`colormap`	Fungsi untuk mengubah *colormap* dari citra yang sedang aktif dijalankan.

Tabel 3. 8 Daftar Fungsi yang Digunakan Pada *Pseudocode* Praproses dan Segmentasi Pembuluh Darah Retina (Bagian Kedua)

No.	Nama Fungsi	Penjelasan
3.	`struct`	Fungsi untuk membuat atau melakukan konversi data ke dalam struktur *array*.
4.	`FrangiFilter2D`	Fungsi untuk menjalankan proses *Frangi filter*.
5.	`sort`	Fungsi untuk mengurutkan nilai dengan jumlah dua atau lebih.
6.	`zeros`	Fungsi untuk membuat matriks yang terdiri dari angka nol.
7.	`disp`	Fungsi untuk menampilkan suatu *array* atau ekspresi.
8.	`Hessian2D`	Fungsi untuk mendapatkan hasil deriviasi orde kedua dari *Hessian matrix*.
9.	`eig2image`	Fungsi untuk memperoleh *eigenvalue* dari citra dua dimensi.
10.	`atan2`	Fungsi untuk menghitung kuadran empat *inverse* tangen dari suatu elemen.
11.	`length`	Fungsi untuk menghitung panjang suatu vektor.
12.	`max`	Fungsi untuk menemukan nilai maksimum dari kumpulan nilai yang ada.
13.	`reshape`	Fungsi untuk mengembalikan nilai *array* dengan elemen yang sama dan dilakukan pengembalian bentuk dalam ukuran tertentu.
14.	`ones`	Fungsi untuk membuat matriks yang terdiri dari angka satu.

Tabel 3. 9 Daftar Fungsi yang Digunakan Pada *Pseudocode* Praproses dan Segmentasi Pembuluh Darah Retina (Bagian Ketiga)

No.	Nama Fungsi	Penjelasan
15.	`ndgrid`	Fungsi untuk melakukan replikasi terhadap vektor jaringan (*grid vector*) dengan menghasilkan koordinat *array*.
16.	`imfilter`	Fungsi untuk melakukan n-dimensi *filter* pada suatu citra.
17.	`abs`	Fungsi untuk mendapatkan nilai absolut dari suatu elemen.
18.	`im2uint8`	Fungsi untuk mengubah citra ke dalam presisi uint8.
19.	`fspecial`	Fungsi untuk menghasilkan *filter* dua dimensi dari tipe tertentu.
20.	`masking`	Fungsi untuk menjalankan proses *Masking*.
21.	`size`	Fungsi untuk mendapatkan nilai ukuran (*size*) setiap dimensi dari suatu *array*.
22.	`bsxfun`	Fungsi untuk menghasilkan operasi biner *element-by-element* untuk dua *array* dengan mengaktifkan ekspansi tunggal.
23.	`bradley`	Fungsi untuk menjalankan proses *Bradley thresholding*.
24.	`averagefilter`	Fungsi untuk menjalankan proses *average filter*.
25.	`true`	Fungsi untuk mengonversikan suatu nilai ke dalam *array* logika *ones*.
26.	`bwareaopen`	Fungsi untuk menghilangkan atau menghapus piksel kecil dari citra biner.

Tabel 3. 10 Daftar Fungsi yang Digunakan Pada *Pseudocode* Praproses dan Segmentasi Pembuluh Darah Retina (Bagian keempat)

No.	Nama Fungsi	Penjelasan
27.	`strel`	Fungsi untuk membuat *structuring element* morfologi.
28.	`imclose`	Fungsi untuk menjalankan proses morfologi *close*.

3.1.1 Program Utama

Program utama merupakan program yang memanggil fungsi-fungsi lain untuk menjalankan program secara keseluruhan. Proses-proses yang dilakukan dalam algoritma yang digunakan dipisahkan dalam fungsi-fungsi yang. *Pseudocode* program utama ditunjukkan pada Gambar 3.3.

Masukan	Citra retina dalam ruang warna RGB (variabel img)
Keluaran	Citra deteksi pembuluh retina (variabel frangiv)

```
1.  Im ← double(img)
2.  input ← Im(:,:,2)
3.  options ← struct('FrangiScaleRange',[1 10],
4.  'FrangiScaleRatio',2,'FrangiBetaOne',0.5,
5.  'FrangiBetaTwo',15,'verbose',true,'BlackWhi
6.  te',true)
7.  frangiv ← FrangiFilter2D(input,options)
8.  img2 ← im2uint8(frangiv)
9.  meanf ← meanfilter(img2)
    maskedImage ← masking(meanf, circleImage)
    bt ← bradley(maskedImage, [5 5], 25)
10. mo ← bwareaopen(bt, 21)
    se ← strel('square', 2)
    mc ← imclose(mo, se)
```

Gambar 3. 3 *Pseudocode* program utama

3.1.2 *Hessian Matrix* dan *Eigenvalue*

Program *Hessian matrix* merupakan program yang digunakan untuk mendapatkan nilai matriks dari *Hessian matrix* yang telah dijelaskan pada subbab 2.2.1. *Pseudocode* dari *Hessian matrix* ditunjukkan pada Gambar 3.4.

Masukan	Citra retina dalam ruang warna RGB (variabel Im), nilai skala sigma (variabel sigmas)
Keluaran	Nilai deviasi orde kedua Hessian Matrix(variabel Dxx, Dxy, Dyy)

```
1.  if nargin < 2
2.          Sigma ← 1
3.  end
4.  [X,Y] ←
        ndgrid(-round(3*Sigma):round(3*Sigma))
5.  DGaussxx ← 1/(2*pi*Sigma^4)
          * (X.^2/Sigma^2 - 1)
     .* exp(-(X.^2 + Y.^2)/(2*Sigma^2))
6.  DGaussxy ← 1/(2*pi*Sigma^6) * (X .* Y)
     .* exp(-(X.^2 + Y.^2)/(2*Sigma^2))
7.  DGaussyy ← DGaussxx'
8.  Dxx ← imfilter(Im, DGaussxx, 'conv')
9.  Dxy ← imfilter(Im, DGaussxy, 'conv')
10. Dyy ← imfilter(Im, DGaussyy, 'conv')
s
```

Gambar 3. 4 *Pseudocode* program *Hessian Matrix*

Sedangkan program *eigenvalue* merupakan program yang digunakan untuk menghitung nilai *eigenvalue* dari *Hessian matrix* yang telah dijelaskan pada subbab 2.2.2. *Pseudocode* program dari *eigenvalue* ditunjukkan pada Gambar 3.5.

Masukan	Nilai deviasi orde kedua dari Hessian matrix (variabel Dxx, Dxy, dan Dyy)
Keluaran	Nilai eigenvalue (variabel Lambda1 dan Lambda2), Nilai vektor (variabel Ix dan Iy)

```
1.   tmp ← sqrt((Dxx - Dyy).^2 + 4*Dxy.^2)
2.   v2x ← 2*Dxy
3.   v2y ← Dyy - Dxx + tmp
4.   mag ← sqrt(v2x.^2 + v2y.^2)
5.   I ← (mag ~= 0)
6.   v2x(i) ← v2x(i)./mag(i)
7.   v2y(i) ← v2y(i)./mag(i)
8.   v1x ← -v2y
9.   v1y ← v2x
10.  mu1 ← 0.5*(Dxx + Dyy + tmp)
11.  mu2 ← 0.5*(Dxx + Dyy - tmp)
12.  check ← abs(mu1) > abs(mu2)
13.  Lambda1 ← mu1
14.  Lambda1(check) ← mu2(check)
15.  Lambda2 ← mu2
16.  Lambda2(check) ← mu1(check)
17.  Ix ← v1x
18.  Ix(check) ← v2x(check)
19.  Iy ← v1y
20.  Iy(check) ← v2y(check)
```

Gambar 3. 5 *Psuedocode Eigenvalue*

3.1.3 *Frangi Filter*

Program Frangi filter merupakan program yang digunakan untuk mengimplementasikan fungsi Frangi filter yang telah dijelaskan pada subbab 2.2.3. Pada Gambar 3.6 dan Gambar 3.7 ditunjukkan pseudocode dari Frangi filter.

Masukan	*Green channel retina*(variabel masukkan), nilai frangi filter (variabel options)
Keluaran	Citra hasil peningkatan (variabel outIm), matriks skala intensitas maksimal (variabel whatScale), matriks direksi piksel (variabel Direction)

```
1.  sigmas ← options.FrangiScaleRange(1) :
    options.FrangiScaleRatio             :
    options.FrangiScaleRange(2)
2.  sigmas ← sort(sigmas, 'ascend')
3.  beta ← 2*options.FrangiBetaOne^2
4.  c ← 2*options.FrangiBetaTwo^2
5.  ALLfiltered ← zeros([size(Im)
                  length(sigmas)])
6.  ALLangles ← zeros([size(Im)
                  length(sigmas)])
7.  for i←1 to length(sigmas)
8.      if(options.verbose)
9.   disp(['Current Frangi Filter Sigma:'
     num2str(sigmas(i))])
10.     end
11.     [Dxx,Dxy,Dyy] ←
     Hessian2D(Im,sigmas(i))
12.     Dxx ← (sigmas(i)^2)*Dxx
13.     Dxy ← (sigmas(i)^2)*Dxy
14.     Dyy ← (sigmas(i)^2)*Dyy
15.     [Lambda2,Lambda1,Ix,Iy] ←
     eig2image(Dxx,Dxy,Dyy)
16.     angles ← atan2(Ix,Iy)
17.     Lambda1(Lambda1==0) ← eps
18.     Rb ← (Lambda2./Lambda1).^2
19.     S2 ← Lambda1.^2 + Lambda2.^2
```

Gambar 3. 6 *Pseudocode Frangi Filter* (Bagian Pertama)

```
20.  Ifiltered          ←          exp(-Rb/beta)
      .*(ones(size(Im))-exp(-S2/c))
21.      if(options.BlackWhite)
22.          Ifiltered(Lambda1<0) ← 0
23.      else
24.          Ifiltered(Lambda1>0) ← 0
25.      end
26.      ALLfiltered(:,:,i) ← Ifiltered
27.      ALLangles(:,:,i) ← angles
28.  end
29.  if length(sigmas) > 1
30.      [outIm,whatScale] ←
      max(ALLfiltered,[],3)
31.      outIm ← reshape(outIm,size(Im))
32.      if(nargout>1)
33.          whatScale ←
            reshape(whatScale,size(Im))
34.      end
35.      if(nargout > 2)
36.          Direction ← reshape(ALLangles
            ((1:numel(Im))'+(whatScale(:)-
            1)*numel(Im)),size(Im))
37.      end
38.  else
39.      outIm ← reshape(ALLfiltered,size(Im))
40.      if(nargout > 1)
41.          whatScale ← ones(size(Im))
42.      end
43.      if(nargout > 2)
44.          Direction ←
            reshape(ALLangles,size(Im))
45.      end
46.  end
```

Gambar 3. 7 *Pseudocode Frangi Filter* (Bagian Kedua)

3.1.4 *Mean Filter*

Program *cleaning* mengaplikasikan proses *cleaning* yakni menghapus objek pembuluh darah berukuran kecil serta mengisi lubang berukuran kecil. *Pseudocode* ini mengaplikasikan metode yang telah dijelaskan pada subbab 2.2.8. *Pseudocode* proses *cleaning* wavelet ditunjukkan pada Gambar 3.8.

Masukan	Citra retina hasil proses Frangi filter (variabel frangiv)
Keluaran	Citra hasil mean filter (variabel meanf)

```
1.  img2 ← im2uint8(frangiv)
2.  mean ← fspecial('average', [3 3])
3.  meanf ← imfilter(img2, mean)
```

Gambar 3. 8 *Pseudocode Mean Filter*

3.1.5 *Masking*

Program *masking* merupakan program yang digunakan untuk mendapatkan citra hasil penghapusan tepi citra pembuluh darah retina dengan menggunakan *mask* yang telah dijelaskan pada subbab 2.3.2. *Pseudocode* dari *masking* ditunjukkan pada Gambar 3.9 dan Gambar 3.10.

Masukan	Citra retina hasil proses mean filter (variabel meanf), citra mask retina (variabel circleImage)
Keluaran	Citra hasil proses Masking(variabel maskedImage)

```
1.   [rows, columns, numberOfColorBands]
                                        ←
2.   size(meanf)
3.   if numberOfColorBands == 1
4.      maskedImage ← meanf
         maskedImage(~circleImage) ← 0
```

Gambar 3. 9 *Pseudocode Masking* (Bagian Pertama)

```
5.    Else
6.    maskedImage ← bsxfun(@times,
      meanf,cast(circleImage,class(meanf)))
7.     end
```

Gambar 3. 10 *Pseudocode Masking* (Bagian Kedua)

3.1.6 *Bradley Thresholding*

Program *Bradley thresholding* merupakan program yang digunakan untuk mendapatkan citra hasil proses *thresholding* dengan menggunakan *Bradley local image thresholding* yang telah dijelaskan pada subbab 2.4. *Pseudocode* dari *Bradley thresholding* ditunjukkan pada Gambar 3.11.

Masukan	Citra retina hasil proses Masking (variabel maskedImage), nilai window dan kecerahan ([5 5] dan 25)
Keluaran	Citra hasil thresholding dengan menggunakan Bradley Thresholding (variabel bt)

```
1.    image ← maskedImage
2.    numvarargs ← length(varargin)
3.    if numvarargs > 3
       error('myfuns:somefun2Alt:TooManyInputs',
       'Possible parameters are: (image, [m n],
       T, padding)')
4.    end
5.    optargs ← {[15 15] 10 'replicate'
6.    optargs(1:numvarargs) ← varargin
7.    [window, T, padding] ← optargs{:}
8.    image ← double(image)
9.    avrg ← averagefilter(image, window,
10.   padding)
11.   output ← true(size(image))
      output(image <= avrg*(1-T/100)) ← 0
      bt ← output
```

Gambar 3. 11 *Pseudocode Bradlay Thresholding*

3.1.7 Morfologi *Open*

Program *area opening* merupakan program yang digunakan untuk menghasilkan citra dengan penghapusan piksel-piksel yang kecil dengan menggunakan Morfologi *area opening* yang telah dijelaskan pada subbab 2.5.1. *Pseudocode* dari *Bradley thresholding* ditunjukkan pada Gambar 3.12.

`Masukan`	`Citra retina hasil proses thresholding dengan Bradley thresholding (variabel bt)`
`Keluaran`	`Citra hasil morfologi area opening (variabel mo)`
`1. mo ← bwareaopen(bt, 21)`	

Gambar 3. 12 *Pseudocode Open* (Bagian Pertama)

3.1.8 Morfologi *Close*

Program morfologi *close* merupakan program yang digunakan untuk mendapatkan citra hasil proses morfologi *close* yang telah dijelaskan pada subbab 2.5.2. *Pseudocode* dari morfologi *close* ditunjukkan pada Gambar 3.13.

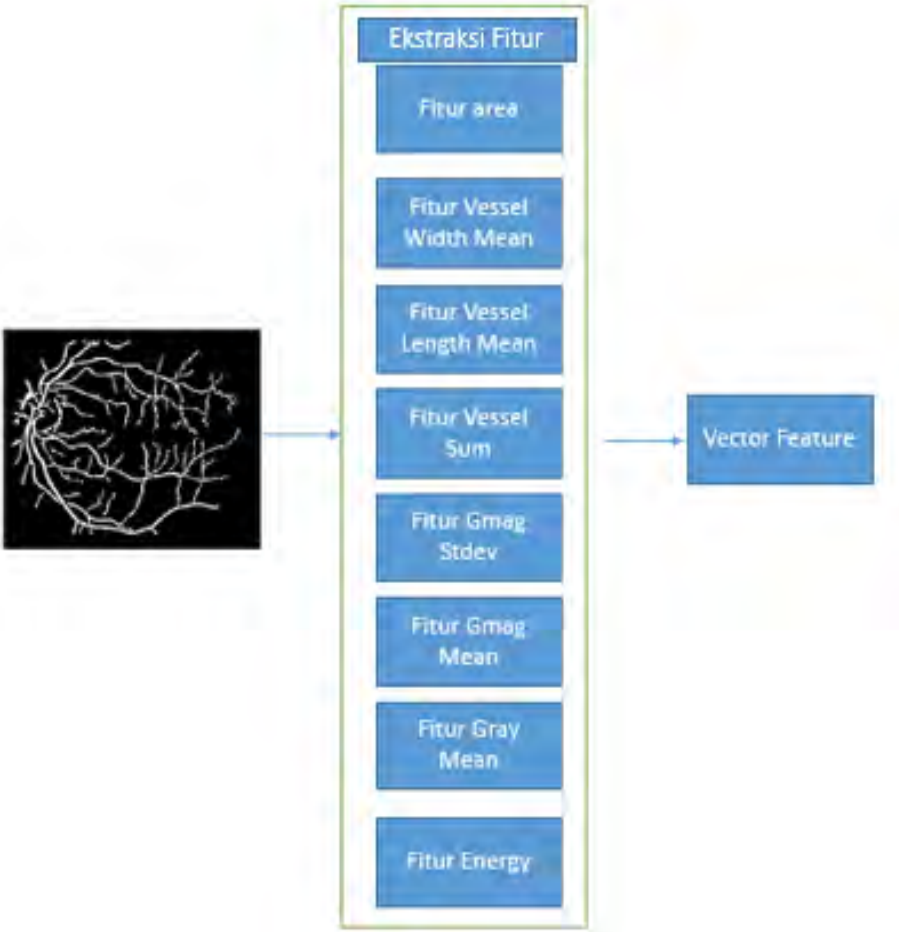
`Masukan`	`Citra hasil Morfologi Area opening (variabel mo)`
`Keluaran`	`Citra hasil Morfologi Close (variabel mc)`
`1. se ← strel(square)`	
`2. mc ← imclose(mo, se)`	

Gambar 3. 13 *Pseudocode close*

3.2. Perancangan Ekstraksi Fitur Citra Segmentasi Retina

Pada subbab ini akan dijelaskan proses ekstraksi fitur dari citra hasil segmentasi. Fitur yang diekstraksi adalah seperti yang dijelaskan pada bab 2 yaitu *area, vessel width mean, vessel length mean, vessel sum, standar deviation gradien magnitude, number of vessel, mean gradient magnitude,* dan *energy*. Penjelasan berupa *psuedocode* serta penjelasan variable atau fungsi yang ada dalam

psuedocode. Diagram utama proses ekstraksi fitur dapat dilihat pada gambar 3.14.



Gambar 3. 14 Gambaran umum proses ekstraksi fitur segmen vaskular retina

Tabel 3. 11 Daftar Variabel yang Digunakan Pada *Pseudocode* Ekstraksi Fitur Segmen Vaskular (Bagian Pertama)

No.	Nama Variabel	Tipe	Penjelasan
1.	`Area`	int	Luas area yang merupakan pembuluh darah pada citra biner hasil segmentasi.
2.	`bw_ segmented`	uint8	Citra biner hasil akhir dari proses segmentasi.
3.	`Cc`	struct	Berupa *struct* yang berisi atribut-atribut hasil dari *connected component analysis*.
4.	`cc.NumObjects`	int	Jumlah objek yang ditemukan dengan analisis *connected component*.
5.	`count`	int	Frekuensi dari piksel untuk setiap nilai intensitas.
6.	`energy`	double	Jumlah intensitas piksel yang merupakan pembuluh darah pada *green channel* dari citra asli.
7	`fixed`	double	Berisi nilaisubset fitur untuk dinormalisasi.
8.	`Gmag`	double	Nilai yang menunjukkan *gradient magnitude*.
9.	`Gdir`	double	Nilai yang menunjukkan *gradieny direction*.
10.	`Im`	uint8	`citra retina dalam ruang warna RGB`
11.	`label`	Uint8	Nilai hasil labelisasi citra retina

Tabel 3. 12 Daftar Variabel yang Digunakan Pada *Pseudocode* Ekstraksi Fitur Segmen Vaskular (Bagian Kedua)

No.	Nama Variabel	Tipe	Penjelasan
12.	`M`	int	Nilai yang menunjukkan jumlah baris citra masukan.
13.	`meangrad`	double	Nilai yang menunjukkan *mean gradient* dari suatu citra.
14.	`meanint`	double	Nilai yang menunjukkan *mean intensity* dari semua piksel yang merupakan pembuluh darah.
15.	`meanintensity`	double	Nilai yang menunjukkan jumlah *mean intensity* dari suatu citra yang telah di-normalisasi.
16.	`Norm`	double	Hasil normalisasi *fitur*.
17.	`Out`	uint8	Citra hasil gabungan antara citra asli retina pada *green channel* dengan citra biner hasil segmentasi.
18.	`stdgrad`	double	Nilai *standard deviation gradient* dari suatu citra.
19.	`subset`	double	Nilai *subset dari suatu feature vector*. Digunakan untuk variable temporer untuk menyiman angka suatu jenis fitur
20.	`res`	double	Berisi nilai rata-rata panjang vessel.

Tabel 3. 13 Daftar Variabel yang Digunakan Pada *Pseudocode* Ekstraksi Fitur Segmen Vaskular (Bagian Ketiga)

No.	Nama Variabel	Tipe	Penjelasan
22.	`vessegm`	int	Nilai yang menunjukkan jumlah objek (segmen) pembuluh darah terkoneksi

Tabel 3. 14 Daftar Fungsi yang Digunakan Pada *Pseudocode* Ekstraksi Fitur Segmen Vaskular (Bagian Pertama)

No.	Nama Fungsi	Penjelasan
1.	`Bwconncomp`	Fungsi untuk menghitung jumlah pada suatu citra.
2.	`pixCount`	Fungsi yang digunakan untuk menghitung fitur *area*.
3.	`energy`	Fungsi yang digunakan untuk menghitung fitur *energy*
4.	`countGradientMag`	Fungsi yang digunakan untuk menghitung fitur *mean gradient*.
5.	`stdgrad`	Fungsi yang digunakan untuk menghitung fitur *standard deviation gradient*.
6.	`imgradient`	Fungsi untuk menghitung *gradient magnitude.*
7.	`featureNormalize`	Fungsi untuk membuat skala 0 sampai 1 untuk suatu vector.
8.	`meangray`	Fungsi yang digunakan untuk menghitung fitur *mean gray level*
9.	`vesselSum`	Fungsi yang digunakan untuk menghitung fitur *vessel segment*.

Tabel 3. 15 Daftar Fungsi yang Digunakan Pada *Pseudocode* Ekstraksi Fitur Segmen Vaskular (Bagian Kedua)

No.	Nama Fungsi	Penjelasan
10.	`gradMag`	Fungsi yang digunakan untuk mendapatkan nilai *gradient magnitude*.
11.	`mean`	Fungsi untuk rata-rata matrix.
12.	`max`	Fungsi untuk menemukan nilai maksimum dari kumpulan data.
13.	`Size`	Fungsi untuk mencari ukuran dari suatu *array*.
14.	`Zeros`	Fungsi untuk membuat matriks yang terdiri dari angka nol.
15.	`Sqrt`	Fungsi untuk menghitung akar kuadrat dari sebuah nilai.
16.	`bwmorph`	Fungsi morfologi untuk *thining* citra

3.2.1 Program Utama

Program utama merupakan program yang memanggil fungsi-fungsi lain untuk menjalankan program secara keseluruhan. Proses-proses yang dilakukan dipisahkan dalam fungsi-fungsi yang berbeda untuk memudahkan pengecekan program setiap prosesnya. *Pseudocode* program utama ditunjukkan pada Gambar 3.15 dan 3.16 merupakan implementasi dari subbab 2.3.

`Masukan`	`Citra segmentasi (variabel bw_segmented),citra retina RGB (variabel im) kanal hijau (imGreen)`
`Keluaran`	`Vektor fitur (variable vector_feature)`

Gambar 3. 15 *Pseudocode* program utama ekstraksi fitur (Bagian Satu)

```
1.  for i ← 1:62
2.  bw_segmented ←pathSegmented(i)
3.  im ← pathIm(i)
4.  imgreen ← pathGreen(i)
5.  area ← pixCount(bw_segmented)
6.  widthMean ← vesselWidthMean(bw_segmented)
7.  lengthMean←vesselLengthMean(bw_segmented)
8.  vesselNum ← vesselSum(bw_segmented)
9.  stdgrad ← countStdev(bw_segmented)
10. gradMag ← countGradientMag(bw_segmented)
11. meangray ← meanGray(imGreen)
12. energy ←
    countGradientMag(bw_segmented,im,
13. imgreen)
14. end
15. featureVector(i) ← normalize([area,
    widthMean, lenthMean, vesselNum, stdgrad,
    gradMag, meangrad, energy])
```

Gambar 3. 16 *Pseudocode* program utama ekstraksi fitur (Bagian Kedua)

3.2.2 Normalisasi Fitur

Program normalisasi fitur digunakan setelah ekstraksi fitur selesai. Program ini akan menjadikan *range* nilai fitur menjadi antara 0 sampai 1. Program ini akan dipanggi pada program utama ekstraksi fitur. *Psuedocode* normalisasi fitur dapat dilihat pada gambar 3.17 dan 3.18.

Masukan	Vector yang berisi hasil ekstraksi fitur (variable feature)
Keluaran	Set fitur yang telah dinormalisasi(variabel norm)

Gambar 3. 17 *Pseudocode* program normalisasi fitur (Bagian Satu)

```
1.  [a,b] ← size(feature)
2.      subset ← zeros
3.      fixed ← zeros()
4.      norm ← zeros(a,b)
5.      for i ← 1:b
6.          for j ← 1:a
7.          subset(j) ← feature(j,i)
8.          end
9.          fixed ←  featureNormalize(subset)
10.         for j← 1:a
11.         norm(j,i) ←  fixed(j)
12.         end
13.     end
```

Gambar 3. 18 *Pseudocode* program normalisasi fitur (Bagian Kedua)

3.2.3 Fitur *Area*

Program fitur *area* merupakan program digunakan untuk menghitung nilai dari fitur *area* yang dijelaskan pada subbab 2.3.1. *Pseudocode* penghitungan *area* ditunjukkan Gambar 3.19.

Masukan	Citra hasil akhir segmentasi (variabel bw_segmented)
Keluaran	Luas area(variabel area)
1. area ← sum(sum(bw_segmented))	

Gambar 3. 19 *Pseudocode* program menghitung *area*

3.2.4 Fitur *Energy*

Program fitur *energy* merupakan program digunakan untuk menghitung nilai dari fitur *energy* yang telah dijelaskan pada subbab 2.3.2. Program ini memiliki inputan berupa citra hasil segmentasi, green channel, serta skala keabuan retina. *Pseudocode* proses penghitungan fitur *energy* ditunjukkan pada Gambar 3.20.

Masukan	Citra vaskular(bw_segmented), green cannel(imgreen), gray scale retina(im)
Keluaran	Nilai energy dari piksel pembuluh darah(variabel energy)

```
1.  out ← bw_segmented & imgreen
2.  for i←1 to m
3.      for j←1 to n
4.          if bw_segmented(i,j)==1
5.          out(i,j) ← im(i,j,3)
6.      end for
7.  end for
8.  energy ← 0
9.  for i←1 to m
10.     for j←1 to n
11.         energy ← energy + out(i,j)
12.     end for
13. end for
```

Gambar 3. 20 *Pseudocode* program menghitung *energy*

3.2.5 Fitur *Mean Gradient*

Program fitur *mean gradient* merupakan program digunakan untuk menghitung nilai dari fitur *mean gradient* yang telah dijelaskan pada subbab 2.3.5. *Pseudocode* proses penghitungan fitur *mean gradient* ditunjukkan pada Gambar 3.21.

Masukan	Segmen vaskular (bw_segmented)
Keluaran	Nilai mean gradient dari piksel pembuluh darah (meanGradientMagnitude)

```
1.  [Gmag, ~] ← imgradient(bw_segmented)
2.  meanGradientMagnitude ← mean(mean(Gmag))
```

Gambar 3. 21 *Pseudocode* program menghitung *mean gradient*

3.2.6 Fitur *Standard Deviation Gradient*

Program fitur *standard deviation gradient* merupakan program digunakan untuk menghitung nilai dari fitur *standard deviation gradient* yang telah dijelaskan pada subbab 2.3.4. *Pseudocode* proses penghitungan fitur *standard deviation gradient* ditunjukkan pada Gambar 3.22.

Masukan	Green channel image (bw_segmented)
Keluaran	Nilai standard deviation gradient dari piksel pembuluh darah (variabel stdgrad)

```
1.  [Gmag, ~] ← imgradient(bw_segmented)
2.  [m,n] ← size(Gmag)
3.  stdgrad ← 0
4.  for i←1 to m
5.      for j←1 to n
6.          stdgrad ← stdgrad + ((Gmag(i,j)-
7.          meangrad)^2)
8.      end for
9.  end for
10. stdgrad ← sqrt(stdgrad/(m*n))
```

Gambar 3. 22 *Pseudocode* program menghitung *standard deviation gradient* (bagian pertama)

3.2.7 Fitur *Mean Gray Level*

Program fitur *mean gray level* merupakan program digunakan untuk menghitung rata-rata grey level sebuah citra 2.3.5. *Pseudocode* proses penghitungan fitur ditunjukkan pada Gambar 3.23.

Masukan	Citra green chanel (imgreen)
Keluaran	Mean gray level(meanGradient)

```
1.  meanGray ← mean(mean(imGreen));
```

Gambar 3. 23 *Pseudocode Mean Gray Level.*

3.2.8 Fitur *Vessel Segment*

Program fitur *vessel segment* merupakan program digunakan untuk menghitung nilai dari fitur *vessel segment*. *Pseudocode* proses penghitungan fitur *vessel segment* ditunjukkan pada Gambar 3.24 dan merupakan implementasi subbab 2.3.7.

Masukan	Citra hasil segmentasi (variabel bw_segmented)
Keluaran	citra hasil segmentasi (variabel vessegm)
1. cc ← bwconncomp(bw_segmented) 2. vessegm ← cc.NumObjects	

Gambar 3. 24 *Pseudocode Vessel Segment*

3.2.9 Fitur *Vessel Length Mean*

Program fitur *vessel length mean* merupakan program digunakan untuk menghitung nilai dari fitur *vessel length mean* yang telah dijelaskan pada subbab 2.3.8. *Pseudocode* proses penghitungan fitur *vessel length mean* ditunjukkan pada Gambar 3.25 dan 3.26.

Masukan	Citra hasil segmentasi (variabel bw_segmented)
Keluaran	Nilai rata rata panjang vessel (variabel res)
1	cc ← bwconncomp(bw_segmented)
2	num ← cc.NumObjects
3	label ← labelMatrix(cc)
4	luas ← 1:num
5	length ← 1:num
6	width ← 1:num
7	thin ← bwmorph(label)
8	[l,k] ←size(bw_segmented)

Gambar 3. 25 **.** *Pseudocode Vessel Length Mean* (Bagian Pertama)

9	For y ← 1:k
10	For x ← 1:l
11	If bw_segmented(x,y)>0 and thin(x,y)>0
12	length(label(x,y)) ← length (label(x,y))+1
13	end
14	end
15	end
16	res = mean(length)

Gambar 3. 26 *Pseudocode Vessel Length Mean* (Bagian Kedua)

3.2.10 Fitur *Vessel Width Mean*

Program fitur *vessel width mean* merupakan program digunakan untuk menghitung nilai dari fitur *vessel width mean* yang telah dijelaskan pada subbab 2.3.7. *Pseudocode* proses penghitungan fitur *vessel width mean* ditunjukkan pada Gambar 3.27 dan 3.28.

Masukan	Citra hasil segmentasi (variabel bw_segmented)
Keluaran	Nilai rata rata lebar vessel (variabel res)

1	cc ← bwconncomp(bw_segmented)
2	num ← cc.NumObjects
3	label ← labelMatrix(cc)
4	luas ← 1:num
5	length ← 1:num
6	width ← 1:num
7	thin ← bwmorph(label)
8	[l,k] ←size(bw_segmented)
9	For y ← 1:k
10	For x ← 1:l

Gambar 3. 27 *Pseudocode* program menghitung *mean vessel width* (Bagian Pertama)

11	`If bw_segmented(x,y)>0`
12	`luas(label(x,y)) ← Luas(label(x,y))+1`
13	`end`
14	`If bw_segmented(x,y)>0 and thin(x,y)>0`
15	`length(label(x,y)) ← length (label(x,y))+1`
16	`end`
	`Width ← zeros;`
17	`for i=1:num`
18	`width(i) ← luas(i)/length(i)`
19	`End`
20	`End`
21	`end`
22	`end`
23	`res = mean(length)`

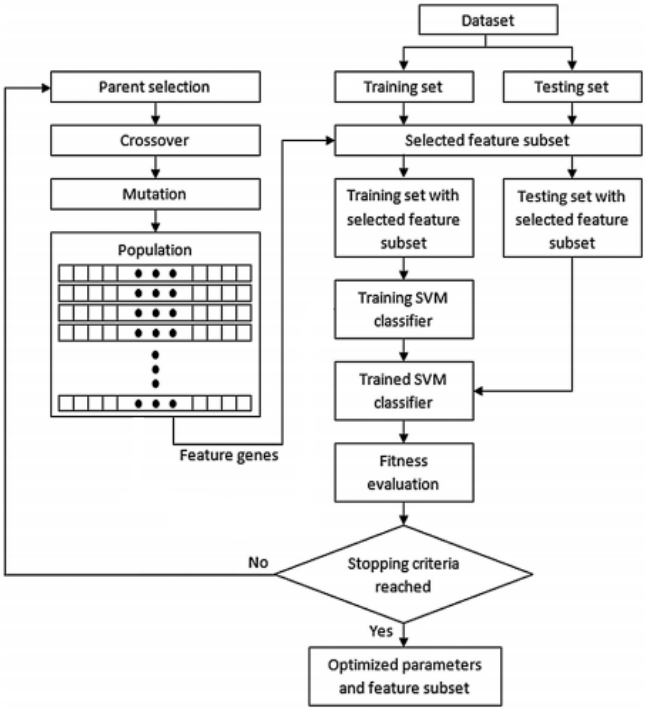
Gambar 3. 28 *Pseudocode* program menghitung mean *vessel width* (Bagian Kedua)

3.3. Perancangan Seleksi Fitur *Genetic Algorithm* dan Klasifikasi *Support Vector Machine*

Genetic Algorithm adalah sebuah algoritma yang bertujuan untuk memecahkan suatu masalah optimasi. Dalam tugas akhir ini GA berperan penting dalam menemukan kombinasi fitur yang direpresentasikan sebagai gen dalam kromosom. GA akan melakukan *crossover* dan mutasi agar kromosom dapat menghasilkan akurasi yang baik sebagai tujuan optimasi.

Proses utama lain yang dilakukan pada akhir Tugas Akhir ini adalah proses klasifikasi segmen vaskular retina sebagai *fitness function*. Proses ini bertujuan untuk melakukan evaluasi terhadap kromosom sehingga muncuk kromosom terbaik. Untuk proses kombinasi GA dan *fitness function* dapat dilihat gambar 3.30.

Kombinasi ini menempatkan SVM sebagai nilai *fitness* dari kromosom. Nilai ini didapat dari *SVMClassify* dengan data *testing* yang telah disediakan. Gambaran umum dari praproses dan segmentasi citra retina digambarkan pada Gambar 3.29.



Gambar 3. 29 diagram alur GA dan SVM

Gambar 3. 30 Desain Kromosom

Program utama ini menggabungkan antara seleksi fitur *GA* dan SVM. Proses diawali dengan inisialisasi populasi awal. Kemudian terjadilah seleksi *parent* untuk *crossover* untuk kemudian dilakukan mutasi. Hasil mutasi akan dihitung akurasinya sebagai *fitness function* menggunakan *SVMClassify*.

Tabel 3. 16 Daftar Variabel yang Digunakan Pada *Pseudocode* Klasifikasi Segmen Vaskular (Bagian Pertama)

No.	Nama Variabel	Tipe	Penjelasan
1.	`acc`	double	Nilai akurasi dari kromosom
2.	`chromosome`	Uint8	*Feature vector* kromosom
3.	`copied`	double	Hasil *copy* dari kromosom
4.	`csOver`	Uint8	Kromosom untuk *crossover*
5.	`cross1`	Uint8	Kromosom hasil *crossover*
6.	`cross2`	Uint8	Kromosom hasil *crossover*
7.	`featureSelect`	double	Variable *training* reduksi fitur setelah dicocokan dengan kromosom
8.	`gaMax`	Int	Iterasi GA maksimal yang dilakukan.
9.	`Group`	Int	Kelas hasil *testing* SVM
10.	`idx`	Uint8	Variable untuk perulangan *crossover*
11.	`init`	int	*Counter* untuk akurasi
12.	`initial`	Uint8	Merupakan variable yang menyimpan set kromosom
13.	`length`	int	Panjang kromosom
14.	`mut`	string	String kromosom setelah mutasi
15.	`num`	int	Jumlah data test
16.	`parent`	double	Variable set fitur awal
17.	`point`	Uint8	Variable random untuk memilih *parent*
18.	`test`	double	Variable testing reduksi fitur setelah dicocokan dengan kromosom
19.	`SVMStruct`	struct	Struct *training* SVM

Tabel 3. 17 Daftar Fungsi yang Digunakan Pada *Pseudocode* Klasifikasi Segmen Vaskular (Bagian Pertama)

No.	Nama Fungsi	Penjelasan
1.	`svmtrain`	Fungsi training svm
2.	`svmclassify`	Fungsi klasifikasi svm
3.	`Bi2de`	Fungsi binary ke desimal
4.	`zeros`	Fungsi deklarai variable dengan size tertentu.
5.	`randi`	Fungsi random integer.
6.	`horzcat`	Fungsi penambahan char secara horisontal.
7.	`sortrows`	Fungsi sorting matrix berdasarkan row

3.3.1 Program *Crossover*

Program ini merupakan tahap awal dari proses *genetic algorithm*. Program ini akan menghasilkan kromosom initial beserta nilai akurasinya. *Psuedocode* ini merupakan penjabaran dari subbab 2.4.1 dapat dilihat pada gambar 3.31.

`Masukan`	`kromosom untuk disilangkan (chromosome1, chrmosome2)`
`Keluaran`	`Variable kromosom hasil(initial)`

```
1.  point ← randi(1,7)
2.  for i←i:8
3.  if i<= point
4.          initial(1,i) ←chromsome2(i)-48;
5.          initial(2,i) ←chromsome1(i)-48;
6.      else
7.          initial(2,i) ←chromsome2(i)-48;
8.          initial(1,i) ←chromsome1(i)-48;
9.   end end
```

Gambar 3. 31 *Psuedocode Crossover*

3.3.2 Program *Mutation*

Program ini merupakan tahap mutasi dari proses *genetic algorithm*. Program ini secara random mengubah nilai kromosom pada gen tertentu. Proses ini memiliki input kromosom yang akan dimutasi. *Psuedocode* ini merupakan penjabaran dari subbab 2.4.1 dan dapat dilihat pada gambar 3.32.

Masukan	Kromosom untuk mutasi (variable chromosome)
Keluaran	Variable kromosom hasil(initial)

```
1.  point ← randperm(8)
2.  for i=1:2
3.  if(chromosome(point(i))==1)
4.      chromosome(point(i)) ←0
5.  else
6.      chromosome(point(i)) ←1
7.  end
8.  end
9.  initial = chromosome;
10. end
```

Gambar 3. 32 *Psuedocode Mutation*

3.3.3 Program *CopyChromosome*

Copychromosome adalah sebuah fungsi untuk menyalin fitur asal menjadi bentuk fitur yang disesuaikan dengan kromosom. Fungsi ini dipanggil saat kromosom akan dilakukan penilaian terhadap *fitness function. Psuedocode* ini merupakan penjabaran dari subbab 2.4.1 dan dapat dilihat pada gambar 3.33 dan 3.34.

Masukan	Kromosom (variable chromosome) dan set fitur (variable parent)
Keluaran	Set fitur baru(variabelcopied)

Gambar 3. 33 *Psuedocode Copychromosome* (Bagian Pertama)

1	length←0
2	for i←1:8
3	if(chromosome(i)==1)
4	length←length+1
5	copied ← zeros(1,length)
6	[a, b] ← size(parent)
7	for j←1:a
8	x←1
9	for i←1:8
10	if(chromosome(i)==1)
11	copied(j,x) ←parent(j,i)
12	x←x+1

Gambar 3. 34 *Psuedocode Copychromosome* (Bagian kedua)

3.3.4 Program toStr

Program ini digunakan sebagai program pembantu untuk mengkonversi fitur biner menjadi string biner. Program ini digunakan dalam program utama *genetic algorithm*. *Psuedocode* program ini dapat dilihat pada gambar 3.35.

Masukan	Kromosom dalam bentuk vektor biner (variable chromosome)
Keluaran	String biner kromosom(str)

1	str ← ''
2	for i←1:8
3	if(chromosome(i)==1)
4	str ←horzcat(temp,'1')
5	else
6	str ←horzcat(temp,'0')
7	End end

Gambar 3. 35 *Psuedocode toStr*

3.3.5 Program *Generate Initial Generation*

Program ini merupakan tahap awal dari proses *genetic algorithm*. Program ini akan menghasilkan kromosom initial beserta nilai akurasinya. *Psuedocode* ini merupakan penjabaran dari subbab 2.4.1 dan dapat dilihat pada gambar 3.36.

Masukan	-
Keluaran	Variable kromosom awal (initial)

```
1.  Initial ← zeros(256,9)
2.  used ← zeros(256,1)
3.  for i ← i:100
4.  x←randi([1,255])
5.  used(x+1) ← 1;
6.  initial(i,1:8) ← dec2bin(x,8)(1,8)
7.  end
8.  for i ← i:100
9.  featureSelect ← copyChromosome(initial(i))
10. Test ← copyChromosome(initial(i))
11. SVMStruct ← svmtrain(featureSelect,label)
12. Group ← svmclassify(SVMStruct,Test)
13. For i ←1:num
14. If Group(i) == label(i)
15. Init←init+1;
16. end
17. end
18. acc ← (init/num)*100
19. initial(i, 9)←acc
20. end
21. initial ← sortrows(initial,-9)
```

Gambar 3. 36 *Pseudocode* program *Initial generation*

3.3.6 Program Utama *Genetic Algorithm* dan SVM

Program utama ini berisikan GA yang telah digabung dengan SVM. Dalam program ini akan dipanggil semua fungsi yang telah didefinisikan sebelumnya. Proses diawali dengan mengambil hasil dari pembuatan generasi awal kemudian

dilakukan proses crossover dan mutasi . *Psuedocode* ini merupakan penjabaran dari subbab 2.4.1 dan dapat dilihat pada gambar 3.37.

Masukan	Kromosom generasi awal (variable initial)
Keluaran	Kromosom akhir (variable initial)

```
1.  gaMax ← 20
2.  for i ← 1:gaMax
3.  initial ← sortrows(initial,-9)
4.  point ←randperm(20)
5.  for idx ←1:4
6.  csOver(idx)=bi2de(initial(point(idx),1:8))
7.  end
8.  cross1 ← crossover(csOver(1), csOver(2))
9.  cross2 ← crossover(csOver(3), csOver(4))
10. mut(1) ← mutation(cross1(1))
11. mut(2) ← mutation(cross1(2))
12. mut(3) ← mutation(cross2(1))
13. mut(4) ← mutation(cross2(2))
14. initial(end+1:4,1:8) ← mut(1:4,1:8)
15. for
16. featureSelect ← copyChromosome(initial(i))
17. Test ← copyChromosome(initial(i))
18. SVMStruct ← svmtrain(featureSelect,label)
19. Group ← svmclassify(SVMStruct,Test)
20. For i ←1:num
21. If Group(i) == label(i)
22. Init←init+1;
23. End end
24. acc ← (init/num)*100
25. initial(i, 9)←acc
26. end end
```

Gambar 3. 37 *Pseudocode* program utama GA

[Halaman ini sengaja dikosongkan]

BAB IV
IMPLEMENTASI

Pada bab ini akan dibahas mengenai implementasi yang dilakukan berdasarkan rancangan yang telah dijabarkan pada bab sebelumnya. Sebelum penjelasan implementasi akan ditunjukkan terlebih dahulu lingkungan untuk melakukan implementasi.

4.1. Lingkungan Implementasi

Lingkungan implementasi yang akan digunakan untuk melakukan implementasi adalah program MATLAB 8.1.0 (R2014b) yang dipasang pada sistem operasi Windows 8.

4.2. Implementasi

Pada bagian ini akan dijelaskan implementasi setiap subbab yang terdapat pada bab sebelumnya yaitu bab perancangan perangkat lunak. Bagian implementasi ini juga akan dibagi menjadi 3 bagian, yaitu bagian praproses dan segmentasi citra retina, bagian ekstraksi fitur segmen vaskular retina, dan bagian klasifikasi segmen vaskular normal dan abnormal.

4.2.1. Implementasi Pemindaian Retina Mata

Implementasi ini merupakan bagian dari implementasi *pseudocode* pada subbab 3.1.1. Implementasi dalam Matlab ditunjukkan pada Kode Sumber 4.1.

```
1       Im = double(img);
2       input = Im(:,:,2);
```

Kode Sumber 4. 1 implementasi pemindaian retina

4.2.2. Implementasi Praproses dan Segmentasi Citra Retina

Pada bagian ini akan dijelaskan implementasi dari praproses dan segmentasi citra retina. Implementasi akan dijelaskan mulai dari praproses yang berupa *masking*. Setelah itu dilanjutkan dengan segmentasi pembuluh darah pada citra retina menggunakan *bradley thresholding*.

4.2.2.1 *Hessian Matrix dan Eigenvalue*

Masukan dari program *Hessian matrix* berupa citra asli retina yang berukuran 565 x 584 piksel dengan tipe data uint8. Citra tersebut akan diproses sehingga menghasilkan persamaan *Hessian matrix*. Implementasi ini merupakan implementasi *pseudocode* pada subbab 3.1.2. Implementasi dalam Matlab ditunjukkan pada Kode Sumber 4.2.

```
1  if nargin < 2
2        sigmas = 1;
3  end
4  [X,Y]   = ndgrid(-
   round(3*sigmas):round(3*sigmas));
5  DGaussxx = 1/(2*pi*sigmas^4) *
   (X.^2/sigmas^2 - 1) .* exp(-(X.^2 +
   Y.^2)/(2*sigmas^2));
6  DGaussxy = 1/(2*pi*sigmas^6) * (X .*
   Y)           .* exp(-(X.^2 +
   Y.^2)/(2*sigmas^2));
7       DGaussyy = DGaussxx';
8       Dxx = imfilter(Im,DGaussxx,'conv');
9       Dxy = imfilter(Im,DGaussxy,'conv');
10      Dyy = imfilter(Im,DGaussyy,'conv');
```

Kode Sumber 4. 2 implementasi *Hessian Matrix*

Sedangkan untuk implementasi dari program *eigenvalue,* dilakukan proses pengaplikasian komputasi *eigenvalue* yang dapat dihitung setelah mendapatkan persamaan *Hessian matrix*. Implementasi ini juga merupakan implementasi *pseudocode* pada

subbab 3.1.2. Implementasi dalam Matlab untuk proses *eigenvalue* ditunjukkan pada Kode Sumber 4.3.

```
1  tmp = sqrt((Dxx - Dyy).^2 + 4*Dxy.^2);
2  v2x = 2*Dxy; v2y = Dyy - Dxx + tmp;
3  mag = sqrt(v2x.^2 + v2y.^2); i = (mag ~= 0);
4  v2x(i) = v2x(i)./mag(i);
5  v2y(i) = v2y(i)./mag(i);
6  v1x = -v2y;
7  v1y = v2x;
8  mu1 = 0.5*(Dxx + Dyy + tmp);
9  mu2 = 0.5*(Dxx + Dyy - tmp);
10 check=abs(mu1)>abs(mu2);
11 Lambda1=mu1; Lambda1(check)=mu2(check);
12 Lambda2=mu2; Lambda2(check)=mu1(check);
13 Ix=v1x; Ix(check)=v2x(check);
14 Iy=v1y; Iy(check)=v2y(check);
```

Kode Sumber 4. 3. implementasi *eigenvalue*

4.2.2.2 *Frangi Filter*

Implementasi *Frangi filter* bertujuan untuk memperbaiki kualitas citra retina. Implementasi ini merupakan implementasi *pseudocode* pada subbab 3.1.3. Implementasi dalam Matlab untuk proses *Frangi filter* ditunjukkan pada Kode Sumber 4.5.

```
1  sigmas=options.FrangiScaleRange(1):options.F
   rangiScaleRatio:options.FrangiScaleRange(2);
2  sigmas = sort(sigmas, 'ascend');
3  beta  = 2*options.FrangiBetaOne^2;
4  c     = 2*options.FrangiBetaTwo^2;
5  ALLfiltered=zeros([size(Im)
   length(sigmas)]);
6  ALLangles=zeros([size(Im) length(sigmas)]);
7  for i = 1:length(sigmas),
8      if(options.verbose)
```

Kode Sumber 4. 4 implementasi *Frangi Filter* (bagian pertama)

```
9           disp(['Current Frangi Filter Sigma:
            ' num2str(sigmas(i)) ]);
10      end
11      [Dxx,Dxy,Dyy] = Hessian2D(Im,sigmas(i));
12      Dxx = (sigmas(i)^2)*Dxx;
13      Dxy = (sigmas(i)^2)*Dxy;
14      Dyy = (sigmas(i)^2)*Dyy;
15      [Lambda2,Lambda1,Ix,Iy]
            =eig2image(Dxx,Dxy,Dyy);
16      angles = atan2(Ix,Iy);
17      Lambda1(Lambda1==0) = eps;
18      Rb = (Lambda2./Lambda1).^2;
19      S2 = Lambda1.^2 + Lambda2.^2;
20      Ifiltered = exp(-
            Rb/beta) .*(ones(size(Im))-exp(-
            S2/c));
21      if(options.BlackWhite)
22          Ifiltered(Lambda1<0)=0;
23      else
24          Ifiltered(Lambda1>0)=0;
25      end
26      ALLfiltered(:,:,i) = Ifiltered;
27      ALLangles(:,:,i) = angles;
28  end
29  if length(sigmas) > 1,
30          [outIm,whatScale] =
            max(ALLfiltered,[],3);
31          outIm = reshape(outIm,size(Im));
32          if(nargout>1)
33       whatScale =reshape(whatScale,size(Im));
34          end
35          if(nargout>2)
36              Direction =
            reshape(ALLangles((1:numel(Im))'+(wh
            atScale(:)-1)*numel(Im)),size(Im));
37          end
38       else
```

Kode Sumber 4. 5. implementasi *Frangi* (bagian Kedua)

```
39      outIm = reshape(ALLfiltered,size(Im));
40      if(nargout>1)
41              whatScale = ones(size(Im));
42      end
43      if(nargout>2)
44          Direction =
                reshape(ALLangles,size(Im));
45      end
46  end
```

Kode Sumber 4. 6 implementasi *Frangi* (bagian Ketiga)

4.2.2.3 *Mean Flter*

Implementasi dari program *mean filter* dilakukan dengan tujuan untuk meningkatkan citra segmen dari pembuluh darah. Implementasi ini juga merupakan implementasi *pseudocode* pada subbab 3.1.4. Implementasi dalam Matlab untuk proses *eigenvalue* ditunjukkan pada Kode Sumber 4.7.

```
1       img2 = im2uint8(frangiv);
2       mean = fspecial('average', [3 3]);
3       meanf = imfilter(img2, mean);
```

Kode Sumber 4. 7 implementasi *Mean Filter*

4.2.2.4 *Masking*

Implementasi dari *masking* dilakukan dengan tujuan untuk menghilangkan *background* terhadap citra pembuluh darah retina. Implementasi ini merupakan implementasi *pseudocode* pada subbab 3.1.5. Implementasi *masking* dalam Matlab ditunjukkan pada Kode Sumber 4.8.

```
1       [rows, columns, numberOfColorBands] =
    size(meanf);
2       if numberOfColorBands == 1
3      maskedImage = meanf;
```

Kode Sumber 4. 8 implementasi *Masking* (Bagian Pertama)

```
4       Else
5         maskedImage = bsxfun(@times, meanf,
          cast(circleImage,class(meanf)));
6       End
```

Kode Sumber 4. 9 implementasi *Masking* (Bagian Kedua)

4.2.2.5 *Bradley Thresholding*

Bradley thresholding digunakan untuk melakukan peningkatan citra pembuluh darah retina yang telah dilakukan praproses. Implementasi ini merupakan implementasi *pseudocode* pada subbab 3.1.6. Implementasi dalam Matlab untuk proses *Bradley thresholding* ditunjukkan pada Kode Sumber 4.10.

```
1   numvarargs = length(varargin);
2   if numvarargs > 3
3      error('myfuns:somefun2Alt:
        TooManyInputs', ...
       'Possible parameters are: (image, [m n],
        T, padding)');
4   End
5   optargs = {[15 15] 10 'replicate'};
6   optargs(1:numvarargs) = varargin;
7   [window, T, padding] = optargs{:};
8   image = double(image);
9   avrg = averagefilter(image, window,
    padding);
10  output = true(size(image));
11  output(image <= avrg*(1-T/100)) = 0;
```

Kode Sumber 4. 10 implementasi *Bradley Thresholding*

4.2.2.6 Morfologi *Opening*

Implementasi morfologi *area opening* bertujuan untuk melakukan reduksi terhadap cabang palsu (*spurs)* yang dimiliki citra pembuluh darah retina dengan batas piksel tertentu. Implementasi ini merupakan implementasi *pseudocode* pada

subbab 3.1.7. Implementasi dalam Matlab untuk proses morfologi *area opening* ditunjukkan pada Kode Sumber 4.11.

```
1       mo = bwareaopen(bt, 21);
```

Kode Sumber 4. 11 implementasi *Opening*

4.2.2.7 Morfologi *Closing*

Implementasi morfologi *close* bertujuan untuk melakukan perbaikan citra pembuluh darah retina yang terputus akibat proses sebelumnya dengan batas piksel tertentu. Implementasi ini merupakan implementasi *pseudocode* pada subbab 3.1.8. Implementasi dalam Matlab untuk proses morfologi *clode* ditunjukkan pada Kode Sumber 4.12.

```
1       se = strel('square', 2);
2       mc = imclose(mo, se);
```

Kode Sumber 4. 12 implementasi *Closing*

4.2.3. Implementasi Ekstraksi Fitur Segmen Vaskular Retina

Pada bagian ini akan dijelaskan implementasi dari proses ekstraksi fitur segmen vaskular retina. Implementasi ini bertujuan untuk menghasilkan fitur-fitur yang dapat dijadikan pembeda antara segmen vaskular normal dan abnormal. Fitur-fitur yang digunakan yaitu *area*, *energy*, *mean gradient*, *standard deviation gradient*, *mean intensity*, *intensity variation*, dan *vessel segments*.

4.2.2.1 Program Utama

Program utama bertujuan untuk mengekstraksi semua fitur untuk dijadikan dalam sebuah fitur vektor. Masukan untuk program ini adalah berupa citra biner hasil segmentasi, citra asli, serta citra dalam warna hijau. Output dari program utama ini adalah *vector feature*. Implementasi ini merupakan implementasi *pseudocode* pada subbab 3.2.1 ditunjukkan pada Kode Sumber 4.13 dan 4.14.

```
1   noCitra = 63;
2   num=1;
3   featureIUWT = zeros;
4   feature = zeros(62,8);
5   label=zeros;
6   z=strcat('');
7   class=0;
8   while num < noCitra;
9       if num < 10
10          z=strcat(num2str(num));
11      else
12          z=strcat(num2str(num));
13      end
14      if num <= 39
15          class=1;
16      else
17          class=0;
18      end
19  thePathGreen = strcat('D:\TA\My TA\data set
    retina\DRIVE\dataset\all\g',z,'.PNG');
20  imgreen = mat2gray(imread(thePathGreen));
21  thePathLineOp = strcat('D:\TA\Code-2016-
    06-02\Code\f',z,'.png');
22  lineOp = mat2gray(imread(thePathLineOp));
23  thePathIm = strcat('D:\TA\My TA\data set
    retina\DRIVE\dataset\all\im',z,'.ppm');
24  im = mat2gray(imread(thePathIm));
25  feature(num,1)=pixCount(lineOp);
```

Kode Sumber 4. 13 implementasi *Program Utama* (Bagian satu)

```
26  feature(num,2)=vesselWidthMean(lineOp);
27  feature(num,3)=mean(vesselLengthMean(lineOp));
28  feature(num,4)=vesselSum(lineOp);
29  feature(num,5)=countStdev(lineOp);
30  feature(num,6)=CountGradientMag(lineOp);
31  feature(num,7)=meanGrayCount(imgreen);
32  feature(num,8)=energyCount(lineOp,imgreen,im);
33  label(num)=class;
34  num = num+1;
35  end
36  feature = normal(feature);
37  dlmwrite('featureSet.csv',feature);
38  dlmwrite('label.csv',label);
```

Kode Sumber 4. 14 implementasi *Program Utama* (Bagian kedua)

4.2.2.2 Normalisasi Fitur

Merupakan fungsi tambahan untuk menormalisasi fitur dari setiap kromosom menjadi antara 0 hingga 1 untuk masing-masing gen. Merupakan implementasi dari *psuedocode* subbab 3.2.2.

```
1   [a,b] = size(feature);
2      subset = zeros;
3      fixed=zeros();
4      norm = zeros(a,b);
5      for i=1:b
6          for j=1:a
7          subset(j)=feature(j,i);
```

Kode Sumber 4. 15 implementasi Normalisasi Fitur (Bagian Pertama)

```
8           end
9           fixed = featureNormalize(subset);
10          for j=1:a
11          norm(j,i)=fixed(j);
12          end
13      end
```

Kode Sumber 4. 16 implementasi Normalisasi Fitur (Bagian Kedua)

4.2.2.3 Fitur *area*

Implementasi fitur *area* untuk menemukan *area* pembuluh darah dari citra biner hasil segmentasi. Implementasi ini merupakan implementasi *pseudocode* pada subbab 3.2.3. Implementasi fitur *area* ditunjukkan pada Kode Sumber 4.17.

```
1   area = sum(sum(bw_segmented));
```

Kode Sumber 4. 17 implementasi fitur Area

4.2.2.4 Fitur *energy*

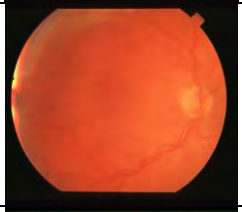
Implementasi fitur *energy* bertujuan untuk menemukan nilai intensitas piksel pembuluh darah pada *green channel* dari suatu citra retina. Masukan untuk program ini adalah berupa citra biner hasil segmentasi serta citra asli retina yang diambil nilai intensitasnya pada *green channel*. Implementasi dalam matlab untuk proses penghitungan fitur *energy* ditunjukkan pada Kode Sumber 4.9. Implementasi ini merupakan implementasi *pseudocode* pada subbab 3.2.4 pada kode sumber 4.18 dan 4.19.

```
1   [m,n]=size(bw_segmented);
2       out = bw_segmented & imgreen;
3   for i = 1:m
4       for j = 1:n
```

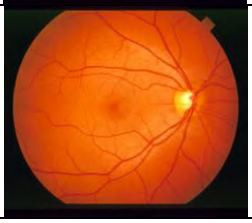
Kode Sumber 4. 18 implementasi *Fitur Energy* (Bagian Pertama)

LAMPIRAN A

Lampiran A. 1 Dataset Citra Retina (Bagian 1)

No	Citra	Nama	Kelas
1		1.ppm	Normal
2		2.ppm	Normal
3		3.ppm	Normal
4		4.ppm	Normal
5		5.ppm	Normal

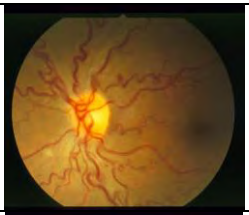
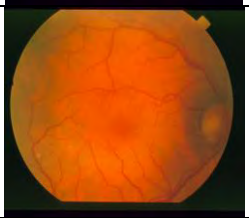
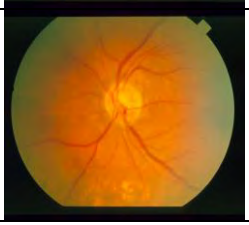
Lampiran A. 2 Dataset Citra Retina (Bagian 2)

No	Citra	Nama	Kelas
6		6.ppm	Normal
7		7.ppm	Normal
8		8.ppm	Normal
9		9.ppm	Normal
10		10.ppm	Normal

Lampiran A. 3 Dataset Citra Retina (Bagian 3)

No	Citra	Nama	Kelas
11		11.ppm	Normal
12		12.ppm	Normal
13		13.ppm	Normal
14		14.ppm	Normal
15		15.ppm	Normal

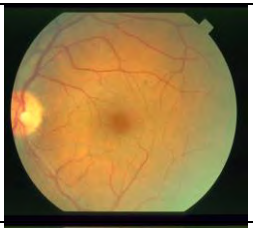
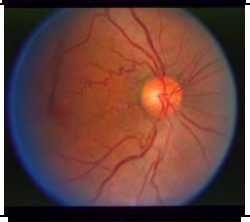
Lampiran A. 4 Dataset Citra Retina (Bagian 4)

No	Citra	Nama	Kelas
16		16.ppm	Normal
17		17.ppm	Normal
18		18.ppm	Normal
19		19.ppm	Normal
20		20.ppm	Normal

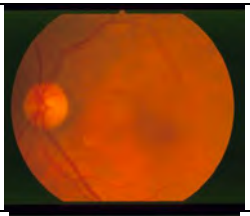
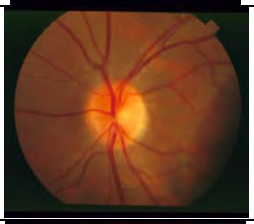
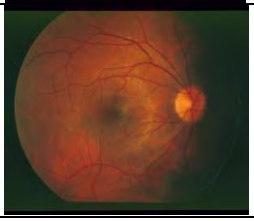
Lampiran A. 5 Dataset Citra Retina (Bagian 5)

No	Citra	Nama	Kelas
21		21.ppm	Normal
22		22.ppm	Normal
23		23.ppm	Normal
24		24.ppm	Normal
25		25.ppm	Normal

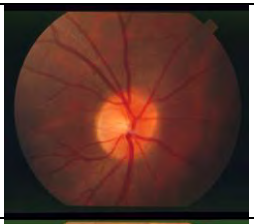
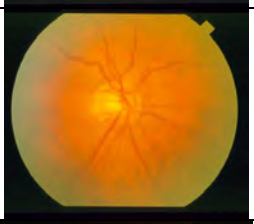
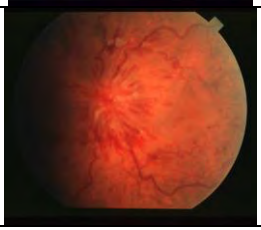
Lampiran A. 6 Dataset Citra Retina (Bagian 6)

No	Citra	Nama	Kelas
26		26.ppm	Normal
27		27.ppm	Normal
28		28.ppm	Normal
29		29.ppm	Normal
30		30.ppm	Normal

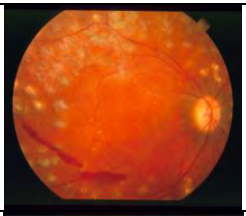
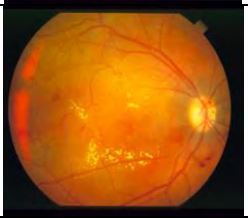
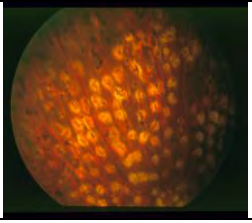
Lampiran A. 7 Dataset Citra Retina (Bagian 7)

No	Citra	Nama	Kelas
31		31.ppm	Normal
32		32.ppm	Normal
33		33.ppm	Normal
34		34.ppm	Normal
35		35.ppm	Normal

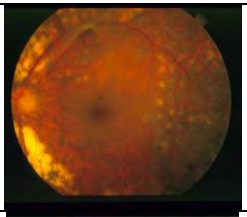
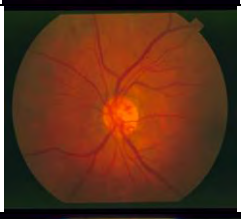
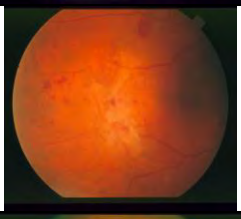
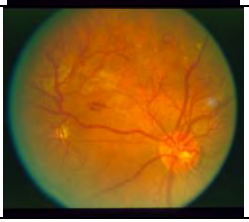
Lampiran A. 8 Dataset Citra Retina (Bagian 8)

No	Citra	Nama	Kelas
36		36.ppm	Normal
37		37.ppm	Normal
38		38.ppm	Normal
39		39.ppm	Normal
40		40.ppm	PDR

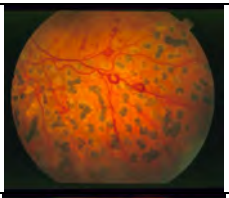
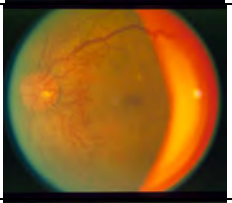
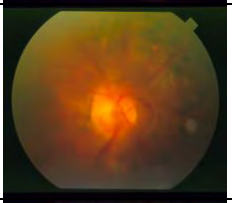
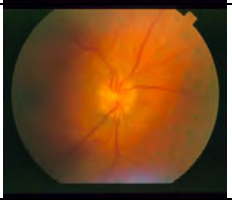
Lampiran A. 9 Dataset Citra Retina (Bagian 9)

No	Citra	Nama	Kelas
41		41.ppm	PDR
42		42.ppm	PDR
43		43.ppm	PDR
44		44.ppm	PDR
45		45.ppm	PDR

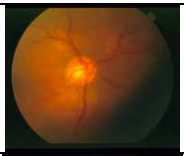
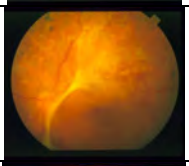
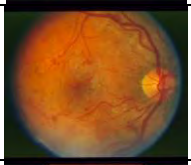
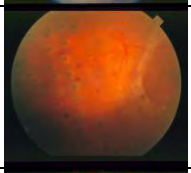
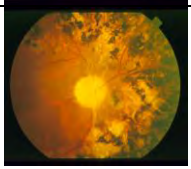
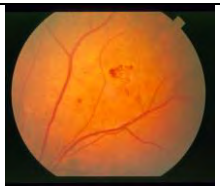
Lampiran A. 10 Dataset Citra Retina (Bagian 10)

No	Citra	Nama	Kelas
46		46.ppm	PDR
47		47.ppm	PDR
48		48.ppm	PDR
49		49.ppm	PDR
50		50.ppm	PDR

Lampiran A. 11 Dataset Citra Retina (Bagian 11)

No	Citra	Nama	Kelas
51		51.ppm	PDR
52		52.ppm	PDR
53		53.ppm	PDR
54		54.ppm	PDR
55		55.ppm	PDR

Lampiran A. 12 Dataset Citra Retina (Bagian 12)

No	Citra	Nama	Kelas
56		56.ppm	PDR
57		57.ppm	PDR
58		58.ppm	PDR
59		59.ppm	PDR
60		60.ppm	PDR
61		61.ppm	PDR

Lampiran A. 13 Data Training dataset 1 (Bagian pertama)

No	Nama	Kelas
1	im1.ppm	Normal
2	im2.ppm	Normal
3	im3.ppm	Normal
4	im4.ppm	Normal
5	im5.ppm	Normal
6	im6.ppm	Normal
7	im7.ppm	Normal
8	im8.ppm	Normal
9	im9.ppm	Normal
10	im10.ppm	Normal
11	im11.ppm	Normal
12	im12.ppm	Normal
13	im13.ppm	Normal
14	im14.ppm	Normal
15	im15.ppm	Normal
16	im16.ppm	Normal
17	im17.ppm	Normal
18	im18.ppm	Normal
19	im19.ppm	Normal
20	im20.ppm	Normal
21	im21.ppm	Normal
22	im22.ppm	Normal
23	im23.ppm	Normal
24	im24.ppm	Normal
25	im25.ppm	Normal
26	im26.ppm	Normal

Lampiran A. 14 Data Training dataset 1 (Bagian Kedua)

No	Nama	Kelas
27	im40.ppm	PDR
28	im41.ppm	PDR
29	im42.ppm	PDR
30	im43.ppm	PDR
31	im44.ppm	PDR
32	im45.ppm	PDR
33	im46.ppm	PDR
34	im47.ppm	PDR
35	im48.ppm	PDR
36	im49.ppm	PDR
37	im50.ppm	PDR
38	im51.ppm	PDR
39	im52.ppm	PDR
40	im53.ppm	PDR
41	im54.ppm	PDR
42	im55.ppm	PDR

Lampiran A. 15 Data Test dataset 1 (Bagian Pertama)

No	Nama	Kelas
1	im27.ppm	Normal
2	im28.ppm	Normal
3	im29.ppm	Normal
4	im30.ppm	Normal
5	im31.ppm	Normal
6	im32.ppm	Normal

Lampiran A. 16 Data Test dataset 1 (Bagian Kedua)

No	Nama	Kelas
7	im33.ppm	Normal
8	im34.ppm	Normal
9	im35.ppm	Normal
10	im36.ppm	Normal
11	im37.ppm	Normal
12	im38.ppm	Normal
13	im39.ppm	Normal
14	im56.ppm	PDR
15	im57.ppm	PDR
16	im58.ppm	PDR
17	im59.ppm	PDR
18	im60.ppm	PDR
19	im61.ppm	PDR

Lampiran A. 17 Data Training dataset 2 (bagian pertama)

No	Nama	Kelas
1	im1.ppm	Normal
2	im2.ppm	Normal
3	im3.ppm	Normal
4	im4.ppm	Normal

Lampiran A. 18 Data Training dataset 2

No	Nama	Kelas
5	im5.ppm	Normal
6	im6.ppm	Normal
7	im7.ppm	Normal
8	im8.ppm	Normal
9	im9.ppm	Normal
10	im10.ppm	Normal
11	im11.ppm	Normal
12	im12.ppm	Normal
13	im13.ppm	Normal
14	im14.ppm	Normal
15	im15.ppm	Normal
16	im16.ppm	Normal
17	im17.ppm	Normal
18	im18.ppm	Normal
19	im19.ppm	Normal
20	Im20.ppm	Normal
21	im40.ppm	PDR
22	im41.ppm	PDR
23	im42.ppm	PDR
24	im43.ppm	PDR
25	im44.ppm	PDR
26	im45.ppm	PDR
27	im46.ppm	PDR
28	im47.ppm	PDR
29	im48.ppm	PDR
30	im49.ppm	PDR
31	im50.ppm	PDR

Lampiran A. 19 Data testing dataset 2

No	Nama	Kelas
1	im21.ppm	Normal
2	im22.ppm	Normal
3	im23.ppm	Normal
4	im24.ppm	Normal
5	im25.ppm	Normal
6	im26.ppm	Normal
7	im27.ppm	Normal
8	im28.ppm	Normal
9	im29.ppm	Normal
10	im30.ppm	Normal
11	im31.ppm	Normal
12	im32.ppm	Normal
13	im33.ppm	Normal
14	im34.ppm	Normal
15	im35.ppm	Normal
16	im36.ppm	Normal
17	im37.ppm	Normal
18	im38.ppm	Normal
19	im39.ppm	Normal
20	im51.ppm	PDR
21	im52.ppm	PDR
22	im53.ppm	PDR
23	im54.ppm	PDR
24	im55.ppm	PDR
25	im56.ppm	PDR
26	im57.ppm	PDR
27	im58.ppm	PDR
28	im59.ppm	PDR
29	im60.ppm	PDR
30	im61.ppm	PDR

Lampiran A. 20 Inisial 20 *run* 1 (5 terbaik)

f1	f2	f3	f4	f5	f6	f7	f8	akurasi
0	1	1	0	0	1	0	0	89.474
0	1	0	1	0	1	1	0	89.474
1	1	0	0	0	0	1	1	89.474
0	0	0	1	0	0	1	1	89.474
1	0	1	1	1	1	1	1	84.211

Lampiran A. 21 Inisial 20 *run* 2 (5 terbaik)

f1	f2	f3	f4	f5	f6	f7	f8	akurasi
1	0	1	0	1	0	1	0	100
1	0	1	1	1	0	1	0	100
1	0	1	0	1	1	1	0	100
1	0	1	1	1	1	1	0	94.737
1	0	1	1	0	1	1	0	94.737

Lampiran A. 22 Inisial 20 *run* 3 (5 terbaik)

f1	f2	f3	f4	f5	f6	f7	f8	akurasi
1	0	0	1	0	1	1	0	94.737
0	1	1	0	1	1	1	0	89.474
0	1	1	0	1	1	0	0	89.474
1	0	0	1	1	1	0	1	89.474
0	0	0	1	1	1	0	0	89.474

Lampiran A. 23 Inisial 20 *run* 4 (5 terbaik)

f1	f2	f3	f4	f5	f6	f7	f8	akurasi
0	0	1	1	0	1	1	0	94.737
1	0	1	0	0	1	0	0	89.474
0	0	1	0	1	0	1	1	84.211
1	1	0	1	0	1	0	0	84.211
0	0	1	1	0	0	1	1	84.211

Lampiran A. 24 Inisial 20 *run* 5 (5 terbaik)

f1	f2	f3	f4	f5	f6	f7	f8	akurasi
1	0	1	1	1	0	1	0	100
1	0	1	0	1	0	1	0	100
0	1	1	1	0	0	1	0	100
0	0	1	1	0	1	1	0	94.737
1	0	0	1	1	1	0	0	94.737

Lampiran A. 25 Inisial 20 *run* 6 (5 terbaik)

f1	f2	f3	f4	f5	f6	f7	f8	akurasi
0	1	1	1	0	0	1	0	100
0	1	1	1	0	1	1	0	100
1	0	1	0	1	1	1	0	100
1	0	1	1	1	1	0	1	94.737
1	0	0	1	1	1	0	0	94.737

Lampiran A. 26 Inisial 20 *run* 7 (5 terbaik)

f1	f2	f3	f4	f5	f6	f7	f8	akurasi
0	1	0	1	0	0	1	1	94.737
0	1	0	1	0	0	1	0	94.737
1	0	0	1	1	1	1	0	89.474
0	1	1	1	0	1	0	0	89.474
0	0	0	1	0	0	1	0	89.474

Lampiran A. 27 Inisial 20 *run* 8 (5 terbaik)

f1	f2	f3	f4	f5	f6	f7	f8	akurasi
0	1	1	1	0	1	1	0	100
1	0	1	1	1	1	0	1	94.737
1	0	0	1	0	1	1	0	94.737
0	1	1	1	0	0	0	0	94.737
1	0	0	1	1	0	1	0	94.737

Lampiran A. 28 Inisial 20 *run* 9 (5 terbaik)

f1	f2	f3	f4	f5	f6	f7	f8	akurasi
1	0	1	1	0	0	1	0	94.737
0	0	0	1	0	1	0	0	89.474
0	0	0	1	0	1	1	0	89.474
1	0	0	0	1	1	1	0	89.474
1	0	1	1	0	1	1	1	84.211

Lampiran A. 29 Inisial 20 *run* 10 (5 terbaik)

f1	f2	f3	f4	f5	f6	f7	f8	akurasi
1	0	1	1	1	0	1	0	100
1	0	1	0	1	0	1	0	100
0	1	1	1	0	1	1	0	100
0	0	1	1	0	0	1	0	94.737
0	1	0	1	0	0	1	0	94.737

Lampiran A. 30 Inisial 50 run 1 (5 terbaik)

f1	f2	f3	f4	f5	f6	f7	f8	akurasi
1	0	1	0	1	0	1	0	100
1	0	1	0	1	1	1	0	100
0	1	0	1	1	0	1	0	94.737
0	1	0	1	1	1	1	0	94.737
1	0	1	1	0	1	1	0	94.737

Lampiran A. 31 Inisial 50 run 2 (5 terbaik)

f1	f2	f3	f4	f5	f6	f7	f8	akurasi
1	0	1	0	1	0	1	0	100
0	1	1	1	0	1	1	0	100
1	0	1	1	1	0	1	0	100
1	0	0	1	1	1	0	0	94.737
0	0	1	1	0	0	1	0	94.737

Lampiran A. 32 Inisial 50 run 3 (5 terbaik)

f1	f2	f3	f4	f5	f6	f7	f8	akurasi
1	0	1	0	1	1	1	0	100
0	1	0	1	1	0	1	0	94.737
1	0	0	1	0	1	1	0	94.737
1	0	1	1	1	1	1	0	94.737
1	0	0	1	1	1	0	0	94.737

Lampiran A. 33 Inisial 50run 4 (5 terbaik)

f1	f2	f3	f4	f5	f6	f7	f8	akurasi
0	0	1	1	0	0	1	0	94.737
0	0	1	1	0	1	1	0	94.737
0	1	0	1	0	0	0	0	94.737
0	1	0	1	0	0	0	0	94.737
0	1	1	1	0	0	0	0	94.737

Lampiran A. 34 Inisial 50 run 5 (5 terbaik)

f1	f2	f3	f4	f5	f6	f7	f8	akurasi
0	1	1	1	0	0	1	0	100
0	1	1	1	0	1	1	0	100
1	0	1	0	1	1	1	0	100
0	1	1	1	1	0	1	0	94.737
1	0	0	1	1	1	0	0	94.737

Lampiran A. 35 Inisial 50run 6 (5 terbaik)

f1	f2	f3	f4	f5	f6	f7	f8	akurasi
0	1	1	1	0	1	1	0	100
0	1	1	1	0	0	1	0	100
0	1	0	1	0	0	0	0	94.737
0	1	1	1	0	0	0	0	94.737
0	1	1	1	1	0	1	0	94.737

Lampiran A. 36 Inisial 50 run 7 (5 terbaik)

f1	f2	f3	f4	f5	f6	f7	f8	akurasi
1	0	1	0	1	0	1	0	100
1	0	1	1	1	1	1	0	94.737
0	1	0	1	1	0	1	0	94.737
1	0	1	1	1	1	0	1	94.737
1	0	0	1	0	0	1	0	94.737

Lampiran A. 37 Inisial 50 run 8 (5 terbaik)

f1	f2	f3	f4	f5	f6	f7	f8	akurasi
1	0	1	0	1	0	1	0	100
0	1	1	1	0	1	1	0	100
0	1	0	1	0	0	0	0	94.737
1	0	1	1	1	1	1	0	94.737
0	1	0	1	0	0	1	0	94.737

Lampiran A. 38 Inisial 50 run 9 (5 terbaik)

f1	f2	f3	f4	f5	f6	f7	f8	akurasi
1	0	1	0	1	0	1	0	100
0	0	1	1	0	0	1	0	94.737
1	0	1	1	1	1	1	0	94.737
0	1	0	1	0	0	1	0	94.737
0	1	1	1	0	0	0	0	94.737

Lampiran A. 39 Inisial 50 run 10 (5 terbaik)

f1	f2	f3	f4	f5	f6	f7	f8	akurasi
0	1	1	1	0	1	1	0	100
0	1	1	1	0	0	1	0	100
1	0	1	0	1	1	1	0	100
0	1	0	1	0	0	0	0	94.737
0	1	0	1	0	0	1	0	94.737

Lampiran A. 40 Inisial 100 run 1 (5 terbaik)

f1	f2	f3	f4	f5	f6	f7	f8	akurasi
0	1	1	1	0	0	1	0	100
1	0	1	1	0	0	1	0	94.737
1	0	1	1	0	1	1	0	94.737
1	0	0	1	1	1	0	0	94.737
0	1	1	1	0	0	0	0	94.737

Lampiran A. 41 Inisial 100 run 2 (5 terbaik)

f1	f2	f3	f4	f5	f6	f7	f8	akurasi
0	1	1	1	0	0	1	0	100
1	0	1	0	1	1	1	0	100
1	0	0	1	1	1	0	0	94.737
1	0	1	1	1	1	1	0	94.737
0	0	1	1	0	0	1	0	94.737

Lampiran A. 42 Inisial 100 run 3 (5 terbaik)

f1	f2	f3	f4	f5	f6	f7	f8	akurasi
1	0	1	1	1	0	1	0	100
1	0	1	1	1	1	0	1	94.737
1	0	0	1	1	0	1	0	94.737
1	0	1	1	1	1	1	0	94.737
0	0	1	1	0	1	1	0	94.737

Lampiran A. 43 Inisial 100 run 4 (5 terbaik)

f1	f2	f3	f4	f5	f6	f7	f8	akurasi
0	1	1	1	0	1	1	0	100
1	0	1	1	1	0	1	0	100
1	0	1	0	1	1	1	0	100
1	0	1	1	0	1	1	0	94.737
0	1	1	1	1	0	1	0	94.737

Lampiran A. 44 Inisial 100 run 5 (5 terbaik)

f1	f2	f3	f4	f5	f6	f7	f8	akurasi
0	1	1	1	0	0	1	0	100
1	0	1	0	1	1	1	0	100
0	0	1	1	0	1	1	0	94.737
0	1	1	1	1	0	1	0	94.737
1	0	1	1	0	1	1	0	94.737

Lampiran A. 45 Inisial 100 run 6 (5 terbaik)

f1	f2	f3	f4	f5	f6	f7	f8	akurasi
0	1	1	1	0	0	1	0	100
1	0	1	1	1	0	1	0	100
1	0	0	1	0	1	1	0	94.737
1	0	1	1	1	1	0	1	94.737
1	0	0	1	1	1	0	0	94.737

Lampiran A. 46 Inisial 100 run 7 (5 terbaik)

f1	f2	f3	f4	f5	f6	f7	f8	akurasi
0	1	1	1	0	1	1	0	100
0	1	1	1	0	0	1	0	100
0	1	1	1	0	0	0	0	94.737
1	0	1	1	0	0	1	0	94.737
1	0	0	1	1	1	0	0	94.737

Lampiran A. 47 Inisial 100 run 8 (5 terbaik)

f1	f2	f3	f4	f5	f6	f7	f8	akurasi
0	1	1	1	0	0	1	0	100
1	0	1	0	1	1	1	0	100
0	1	1	1	1	0	1	0	94.737
1	0	1	1	0	1	1	0	94.737
1	0	1	1	1	1	0	1	94.737

Lampiran A. 48 Inisial 100 run 9 (5 terbaik)

f1	f2	f3	f4	f5	f6	f7	f8	akurasi
1	0	1	1	1	0	1	0	100
1	0	1	0	1	1	1	0	100
1	0	1	1	1	1	1	0	94.737
0	1	0	1	0	0	0	0	94.737
1	0	0	1	1	1	0	0	94.737

Lampiran A. 49 Inisial 100 run 10 (5 terbaik)

f1	f2	f3	f4	f5	f6	f7	f8	akurasi
0	1	1	1	0	1	1	0	100
1	0	1	0	1	1	1	0	100
0	1	1	1	0	0	1	0	100
1	0	1	1	0	0	1	0	94.737
1	0	0	1	1	1	0	0	94.737

Lampiran A. 50 Linear-LS dataset 1:1

f1	f2	f3	f4	f5	f6	f7	f8	akurasi
1	0	0	1	1	0	0	0	93.333
1	0	0	1	1	0	0	1	93.333
0	1	0	1	0	1	0	0	90
1	0	1	1	1	0	0	0	90
0	1	0	1	1	0	1	0	90

Lampiran A. 51 mlp-SMO dataset 1:1

f1	f2	f3	f4	f5	f6	f7	f8	akurasi
1	1	0	1	1	0	0	1	93.333
0	0	1	0	0	0	0	0	90
1	1	0	1	1	0	0	0	86.667
1	1	0	1	0	0	0	0	86.667
1	0	0	0	1	0	0	0	86.667

Lampiran A. 52 quadratic-SMO dataset 1:1

f1	f2	f3	f4	f5	f6	f7	f8	akurasi
0	0	1	0	0	1	0	0	93.333
0	0	1	0	1	0	0	0	90
0	0	1	0	0	1	0	0	90
0	0	1	0	0	1	0	1	90
0	0	1	0	0	0	0	1	86.667

Lampiran A. 53 polynomial-SMO dataset 1:1

f1	f2	f3	f4	f5	f6	f7	f8	akurasi
1	1	1	1	0	0	0	0	93.333
1	0	1	0	0	0	0	0	90
1	1	1	1	0	0	1	0	90
1	1	1	1	1	0	0	0	90
1	1	0	1	1	0	0	0	90

Lampiran A. 54 rbf-SMO dataset 1:1

f1	f2	f3	f4	f5	f6	f7	f8	akurasi
0	1	0	1	0	0	0	0	94.737
0	1	1	1	0	0	0	0	89.474
1	1	0	1	0	0	0	0	89.474
0	1	1	1	0	1	0	0	89.474
1	1	0	1	0	0	0	1	89.474

Lampiran A. 55 linear-SMO dataset 2:1

f1	f2	f3	f4	f5	f6	f7	f8	akurasi
1	1	1	1	0	0	1	1	100
1	1	1	1	0	0	0	1	100
1	1	1	1	0	0	1	0	100
0	1	1	1	1	1	1	0	94.737
0	1	0	1	0	1	1	0	94.737

Lampiran A. 56 mlp-LS dataset 2:1

f1	f2	f3	f4	f5	f6	f7	f8	akurasi
1	0	1	1	1	0	0	1	89.474
1	0	1	1	0	0	0	0	89.474
1	0	0	1	0	0	0	0	89.474
1	0	1	0	0	0	0	0	89.474
1	0	0	1	0	1	0	0	89.474

Lampiran A. 57 quadratic-LS dataset 2:1

f1	f2	f3	f4	f5	f6	f7	f8	akurasi
0	1	1	1	1	0	0	1	94.737
1	1	1	1	1	1	0	1	94.737
1	0	0	1	1	1	0	1	94.737
1	0	1	1	1	1	0	1	94.737
0	0	0	1	1	1	0	1	94.737

Lampiran A. 58 polynomial-SMO dataset 2:1

f1	f2	f3	f4	f5	f6	f7	f8	akurasi
1	0	1	0	1	0	1	0	100
0	1	1	1	0	0	1	0	100
0	1	1	1	1	1	1	0	100
0	1	1	1	0	1	1	0	100
0	1	1	1	1	0	1	0	100

Lampiran A. 59 rbf-QP dataset 2:1

f1	f2	f3	f4	f5	f6	f7	f8	akurasi
0	1	0	1	0	0	0	1	100
1	1	1	1	1	1	1	1	94.737
0	1	1	1	0	1	1	1	94.737
0	1	1	1	0	0	1	1	94.737
1	1	1	1	1	0	1	1	94.737

Lampiran A. 60 QP-polynomial dataset 1:1

f1	f2	f3	f4	f5	f6	f7	f8	akurasi
0	1	1	1	0	0	0	0	93.333
0	1	1	1	0	0	0	0	93.333
0	0	1	0	0	0	0	0	90
0	1	1	1	0	0	0	0	90
1	1	1	1	0	0	1	0	90

Lampiran A. 61 LS-rbf dataset 2:1

f1	f2	f3	f4	f5	f6	f7	f8	akurasi
0	1	0	1	0	0	0	1	100
0	0	1	1	1	0	1	1	94.737
1	1	1	1	1	1	1	1	94.737
0	1	0	1	1	0	1	1	94.737
1	0	1	1	0	0	1	1	94.737

Lampiran A. 62 QP-polynomial dataset 2:1

f1	f2	f3	f4	f5	f6	f7	f8	akurasi
1	0	1	0	1	1	1	0	100
0	1	1	1	0	0	1	0	100
0	1	1	1	0	1	1	0	100
1	0	1	0	1	0	1	0	100
1	0	1	1	1	0	1	0	100

BAB VI
KESIMPULAN DAN SARAN

Bab ini membahas mengenai kesimpulan yang dapat diambil dari hasil uji coba yang telah dilakukan sebagai jawaban dari rumusan masalah yang dikemukakan. Selain kesimpulan, juga terdapat saran yang ditujukan untuk pengembangan perangkat lunak lebih lanjut.

6.1. Kesimpulan

Dari hasil uji coba yang telah dilakukan terhadap pembuatan program praproses dan segmentasi vaskular retina serta ekstraksi fitur dan klasifikasi segmen vaskular retina dapat diambil kesimpulan sebagai berikut:

1. Fitur yang digunakan dalam ekstrksi fitur sudah bagus hal ini terbukti dengan nilai akurasi yang baik. Beberapa kombinasi fitur menghasilkan set fitur yang secara linear terpisahkan sehingga memaksimalkan hasil dari SVM.
2. Metode klsifikasi SVM dapat dikatakan bagus. Hal ini terbukti dengan beberapa konfigurasi SVM yang dijalankan dalam skenario menghasilkan akurasi 100%.
3. Metode seleksi fitur *Genetic Algorithm* dapat dikatakan baik. Hal ini dikarenakan GA dapat memilih fitur yang tepat pada iterasi sedikit dan banyak dengan akurasi yang baik.
4. Sistem GA sangat bergantung pada *initial population* dimana apabila populasi awal baik maka sangat mungkin mendapatkan hasil terbaik.

6.2. Saran

Saran yang diberikan untuk pengembangan aplikasi klasifikasi segmen vaskular retina pada Tugas Akhir ini antara lain:

1. Berdasarkan hasil uji coba, terdapat beberapa data dengan kelas abnormal yang diklasifikasi sebagai data normal. Oleh karena itu dapat dilakukan penambahan fitur yang mampu mengenali segmen vaskular abnormal dengan lebih baik.
2. Untuk meningkatkan hasil uji coba *genetic algorithm*, dapat dilakukan pengembangan metode *genetic algorithm* untuk mempercepat tercapainya kromosom terbaik.

DAFTAR PUSTAKA

[1] R. Weilikala, "Automated Detection of Prolifative Diabetic Retinopathy from Retinal Images".

[2] J.J Staal, M.D. Abramoff, M. Niejmer, M.A. Viergiver, B. van Ginneken, Ridge based vessel segmentation in color images of retina, IEEE, 2004.

[3] Hoover, 5 January 2013. [Online]. Available: ces.clemson.edu/ahoover/stare.

[4] Wikipedia, "Genetic Algorithm," Wikipedia, 20 4 2016. [Online]. Available: en.wikipedia.org/wiki/Genetic_algorithm. [Accessed 5 5 2016].

[5] D.-J. Kroon, "Hessian based Frangi Vesselness filter," Mathworks, 11 June 2009. [Online]. Available: http://www.mathworks.com/matlabcentral/fileexchange/24409-hessian-based-frangi-vesselness-filter. [Accessed 24 December 2014].

[6] HIPR2, "Image Processing Learning Resources," [Online]. Available: http://homepages.inf.ed.ac.uk/rbf/HIPR2/mean.htm. [Accessed 13 11 2015].

[7] R. Jain, R. Kasturi and B. G. Schunck, Machine Vision, McGraw-Hill, 1995.

[8] J. Motl, "Bradley Local Image Thresholding," Mathworks, 18 March 2013. [Online]. Available: http://www.mathworks.com/matlabcentral/fileexchange/40854-bradley-local-image-thresholding. [Accessed 22 October 2015].

[9] "bwareaopen," mathworks, [Online]. Available: http://www.mathworks.com/help/images/ref/bwareaopen.html. [Accessed 22 October 2015].

[10] Wikipedia, "Support Vector Machine," Wikipedia, [Online]. Available: https://en.wikipedia.org/wiki/Support_vector_machine. [Accessed 5 5 2016].

BIODATA PENULIS

Rizkifika Asanul In'am, lahir di Magetan pada tanggal 31 Desember 1993. Penulis telah menempuh pendidikan formal di MI Darul Ulum (2000-2006), SMPN 2 Ngawi (2006-2009), dan SMAN 10 Malang (2009-2012). Pada tahun 2012 penulis diterima di S1 Jurusan Teknik Informatika Institut Teknologi Sepuluh Nopember Surabaya. Di jurusan Teknik Informatika, penulis mengambil bidang minat Komputasi Cerdas dan Visualisasi (KCV). Selama masa kuliah, penulis aktif dalam pada organisasi mahasiswa tingkat jurusan, yaitu Himpunan Mahasiswa Teknik Computer-Informatika (HMTC). Di HMTC penulis pernah menjabat sebagai staff departemen kesejahtraan masyarakat. Penulis juga mengikuti beberapa Latihan Ketrampilan Manajemen Mahasiswa (LKMM) yaitu praTD dan TD.

Selain aktif dalam berorganisasi, penulis juga mengikuti beberapa perlombaan diantaranya PKM-KC didanai sebnyak dua kali pada tahun 2013 dan 2014 serta finalis DevCom 2016. Penulis dapat dihubungi melalui *email*: rizkifika.ai@gmail.com.